VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
KOMPIUTERIJOS KATEDRA

Magistrinis darbas

Dokumentų klasifikavimas semantinių struktūrų pagrindu

Atliko: IIM kurso,
kompiuterinio modeliavimo
grupės studentas
Valentinas Bogdanovičius

Darbo vadovas:
doc. dr. Algimantas Juozapavičius

Vilnius
2006

Turinys

1 Automatinio dokumentų klasifikavimo uždavinys

Didžiąją dalį pasaulinės informacijos sudaro tekstiniai dokumentai. Šimtais metų kaupiamose bibliotekose nuo seno naudojamos informacijos paiešką palengvinančios priemonės: klasifikavimas pagal temą, rūšiavimas pagal autorių, pavadinimą ir t.t. Tačiau toks klasifikavimas reikalauja nemažų žmogiškų resursų ir laiko. Tekstinių dokumentų perkėlimas į kompiuterius žadėjo šios problemos sprendimą, tačiau augant duomenų kiekiams pasirodė, jog trivialūs informacijos paieškos būdai neduoda reikiamų rezultatų nei pagal paieškos laiką, nei kas svarbiau, pagal gaunamų rezultatų prasmę. Problema pasidarė kritinė išplitus Internetui, kai tekstinių duomenų kiekis tapo nebeaprėpiamas, o įprastiniai sintaksiniai paieškos metodai nepateikia prasmingų rezultatų.

Elektroninių tekstinių dokumentų semantinės analizės ir klasifikavimo problemoms spręsti naudojami metodai, kuriuos galima priskirti kompiuterinės lingvistikos (angl. *computational linguistics*), natūralios kalbos apdorojimo (angl. *natural language processing*), tekstų kasybos (angl. *text-mining*) kompiuterijos mokslo šakoms. Yra pasiūlyta nemažai sprendimų kaip reikia įvertinti konkretų dokumentą ir kaip pasirinkti jam labiausiai tinkančią kategoriją. Klasifikavimo metodų pagrindą parastai sudaro statistinė tekstų analizė, dokumentų palyginimai (nagrinėjama šaltinyje [RL00]), algoritmų „apmokymas“ pateikiant „teisingus“ ir „neteisingus“ duomenis, stochastiniai principai. Pritaikius specifinius metodų algoritmus dokumentai yra priskiriami vienai ar kelioms iš anksto apibrėžtoms kategorijoms. Jos gali būti pateikiamos kaip tarpusavyje nesusijusių elementų sąrašas, arba turinčiose hierarchiją semantinėse struktūrose, tokiose kaip taksonomijos (pagal [RD03] taksonomija (angl. *taxonomy*) – kategorijų struktūra; kontroliuojamas žodynas, surūšiuotas pagal sąvokų hierarchiją), tezaurai, semantinės gardelės, ontologijos ir pan. Tokių metodų trūkumu galima įvardinti išankstinę būtinybę apibrėžti kategorijas, dažnai sutinkamas poreikis „mokyti“ algoritmą pavyzdiniais duomenimis.

Tyrimo tikslas – sukurti sistemą, kuri lietuvių kalba parašytus tekstinius dokumentus klasifikuotu pagal dinamiškai generuojamą paprastą semantinę struktūrą, be išankstinio kategorijų aprašo ir algoritmo „apmokymo“.

Tyrimo pradžioje apžvelgti jau žinomi ir naudojami klasifikavimo metodai, įvertinti jų privalumai ir trūkumai. Žinant, jog klasifikavimui bus naudojami lietuviški tekstai, pasiūlyti algoritmai lietuviškų žodžių analizei (žodžio šaknies nustatymui, vienaskaitos vardininko radimui). Aprašytos duomenų struktūros (grafai), kurios gali būti naudojamos

dokumentų semantiniams vaizdams sudaryti. Rasti praktiniai jų sudarymo metodai, aprašytas jų integravimo į vientisą klasifikavimo struktūrą algoritmas. Pasiūlytas algoritmas kategorijų medžio sudarymui iš gautų dokumentų semantinių vaizdų.

Aukščiau išvardintų algoritmų ir metodų pagrindu, realizuotos dokumentų analizės bei integravimo programinės priemonės. Gautųjų duomenų analizei bei eksperimentų atlikimui sukurti papildomi programiniai moduliai. Paruošta bandomoji Web-sąsaja, suteikianti vartotojams dokumentų paieškos funkcionalumą.

Su keliomis pradinių duomenų aibėmis (kurių pagrindą sudarė transporto įstatymai) atlikta daugelis eksperimentų, kurie parodė dokumentų paieškos, tarpusavio palyginimo bei suradimo pagal fragmentą, galimybes. Eksperimentų pagrindu suformuluotos išvados, teigiančios, jog aprašytieji algoritmai ir metodai gali būti panaudoti taikomosiose dokumentų valdymo sistemose.

2 Tekstų klasifikavimo metodai

Formalus tekstų klasifikavimo problemos apibrėžimas pateikiamas šaltinyje [SJ03]: tai yra matematinio modelio, galinčio reprezentuoti teksto semantiką ir būti naudojamu pakartotiniams palyginimams, paieška. Tyrimų tikslas yra algoritmo, galinčio priskirti tekstą tam tikrai kategorijai su maksimaliu tikslumu išvengiant priskyrimo daugeliui kategorijų, radimas.

2.1 Bayes klasifikatorius

Bayes klasifikatorius (angl. *Bayes classifier*) yra paprastas tikimybinis klasifikavimo metodas. Jis priskiria labiausiai tinkamą klasę dokumentui, kuris yra aprašomas kaip savybių vektorius. Metode naudojamos savybių nepriklausomumo prielaidos, kurios dažnai neturi pagrindo realiame gyvenime (dėl šios priežasties dažnai vadinamas „naiviu“ (angl. *naive*)). Nežiūrint to, šis metodas pakankamai sėkmingai naudojamas praktikoje, dažnai pagal efektyvumą konkuruodamas su metodais naudojančiais įmantresnę techniką.

Norint matematiškai apibrėžti Bayes klasifikatorių, reikia pateikti nepriklausomų savybių tikimybinį modelį, kuris šaltinyje [WP01] aprašomas taip:

$$P(C \mid F_1,...,F_n),$$

kur *C* klasių aibė, o $F_1,\ldots,F_n$ savybių kintamieji. Problema yra galimas didelis savybių skaičius, kuris daro tikimybėmis paremtą klasifikavimą neįmanomu. Todėl pasinaudojus Bayes‘o teorema šis modelis performuluojamas:

$$P(C \mid F_1,..., F_n) = \frac{P(C)P(F_1,..., F_n \mid C)}{P(F_1,..., F_n)}.$$

Praktikoje įdomus yra tik šios trupmenos skaitiklis, kadangi vardiklis yra nepriklausomas nuo C ir savybių $F_1,\ldots,F_n$ reikšmės yra duotos, taigi gali būti laikomas konstanta. Skaitiklis yra ekvivalentiškas jungtinės tikimybės modeliui (angl. *joint probability*)

$$P\ (C\ ,\ F_1\ ,...,\ \ F_n\ )\ ,$$

kuris gali būti perrašytas pakartotinai naudojant sąlyginės tikimybės (angl. *conditional probability*) apibrėžimą:

$$
\begin{aligned}
& P(C, F_1, ..., F_n) \\
& = P(C)P(F_1, ..., F_n \mid C) \\
& = P(C)P(F_1 \mid C)P(F_2, ..., F_n \mid C, F_1) \\
& = P(C)P(F_1 \mid C)P(F_2 \mid C, F_1)P(F_3, ..., F_n \mid C, F_1, F_2) \\
& = P(C)P(F_1 \mid C)P(F_2 \mid C, F_1)P(F_3 \mid C, F_1, F_2)P(F_4, ..., F_n \mid C, F_1, F_2, F_3) \\
& \text{M}
\end{aligned}
$$

Panaudojus iškeltas savybių nepriklausomumo prielaidas, $\forall\ i, j : i \neq j\ (i,j \in [1,\ldots,n])$ gauname:

$$P(F_i \mid C, F_j) = P(F_i \mid C),$$

iš ko seka, kad:

$$
\begin{aligned}
& P(C, F_1, ..., F_n) \\
& = P(C)P(F_1 \mid C)P(F_2 \mid C)P(F_3 \mid C)\text{K} \\
& = P(C)\prod_{i=1}^{n} P(F_i \mid C)
\end{aligned}
$$

Tai reiškia, kad panaudojus sąlyginis pasiskirstymas virš klasės *C* gali būti išreikštas taip:

$$P(C \mid F_1, ..., F_n) = \frac{1}{Z} P(C)\prod_{i=1}^{n} P(F_i \mid C),$$

kur *Z* yra konstanta, priklausoma tik nuo $F_1,\ldots,F_n$ reikšmių.

Dabar galime matematiškai apibrėžti Bayes klasifikatorių. Paprasčiausia ir dažnai naudojama klasifikavimo prielaida yra žinoma kaip *maximum a posteriori* arba MAP taisyklė. Jos klasifikatorius atrodo taip:

$$classify(f_1, ..., f_n) = \arg\max_c P(C = c)\prod_{i=1}^{n} P(F_i = f_i \mid C = c).$$

Pailiustruosime Bayes klasifikatoriaus veikimo principą pavyzdžiu. Tarkime, turime klasifikuoti tekstus į mokslinės ir grožinės literatūros klases. Lentelėje pateikiame pagal klasifikavimo savybes suskirstytus pradinius duomenis:

	Tekste sutinkamas žodis ‘aš’	Tekste sutinkami skaičiai	Tekste sutinkamos formulės	Tekstas priklauso klasei
1	Nesutinkamas	Dažnai	Dažnai	Mokslinė literatūra
2	Retai	Retai	Retai	Mokslinė literatūra
3	Nesutinkamas	Dažnai	Nesutinkamos	Mokslinė literatūra
4	Retai	Dažnai	Nesutinkamos	Grožinė literatūra
5	Retai	Nesutinkami	Nesutinkamos	Grožinė literatūra
6	Dažnai	Retai	Retai	Grožinė literatūra

Algoritmui keliama užduotis nustatyti, kokiai klasei priklauso tekstas su savybėmis:

	Tekste sutinkamas žodis ‘aš’	Tekste sutinkami skaičiai	Tekste sutinkamos formulės	Tekstas priklauso klasei
	Retai	Retai	Nesutinkamos	?

Pirmame žingsnyje paskaičiuojame tikimybę kiekvienai klasei, remiantis pradinių duomenų pasiskirstymu:

*P(Grožinė) = (2 / 3) * (1 / 3) * (2 / 3) = 4 / 27*

*P(Mokslinė) = (1 / 3) * (1 / 3) * (1 / 3) = 1 / 27*

Atsižvelgiame į kiekvienos klasės pasitaikymo tikimybę pagal pradinius duomenis:

*P(Grožinė) = (4 / 27) * (3 / 6) = 4 / 54*

*P(Mokslinė) = (1 / 27) * (3 / 6) = 1 / 54*

Iš gauto rezultato darome išvadą, kad tekstas su nurodytomis savybėmis priklauso grožinės literatūros klasei, kadangi tai klasei apskaičiuota tikimybė yra didžiausia.

Klasifikavus kiekvieną naują objektą jis gali būti įtraukimas į pradinių duomenų sąrašą, taip didinant teisingo klasifikavimo tikimybę ateityje. Tačiau augantis duomenų skaičius reiškia ir didesnį poreikį skaičiavimo resursams.

2.2 K – artimiausio kaimyno metodas

K-artimiausio kaimyno metodas (angl. *k-nearest neighbor, k-NN*) yra gerai žinomas ir priklauso vektorinių metodų grupei. Jo esmė yra dokumentų ir sąvokų atvaizdavimas specialioje vektorinėje erdvėje. Vektorinės erdvės dimensijos atitinka naudojamų klasifikacijai raktinių požymių aibę. Prieš pradedant klasifikavimą, algoritmui pateikiami objektai, kurie turi raktinių požymių. Šie mokymo objektai yra pasirenkami iš anksto apibrėžtoms kategorijoms. Tada kiekvienam naujam klasifikuojamam objektui vykdomas paprastas algoritmas:

- objektui randami *k* artimiausių pagal tam tikrą (euklidinį, kosinusinį ir pan.) atstumą objektų iš apmokymo aibės;
- ištyrus gautus *k* kaimynų nustatoma, kokia kategorijai jų priklauso daugiausiai, ir tai kategorijai yra priskiriamas klasifikuojamas objektas.

Matematiškai šis modelis pagal [ZD02] aprašomas taip: naujo dokumento *d* priskyrimas kategorijai c_i jei c_i turi didžiausią panašumo rodiklį (angl. *similarity score*) tarp visų kategorijų. Panašumo rodiklis dokumentui d kategorijai c_i apskaičiuojamas pagal formulę:

$$s(d, c_j) = \sum_{d_i \in k-NN} sim(d, d_i) y(d_i, c_j),$$

kur *sim(d,d_i)* – panašumas tarp dokumento *d* ir mokomojo dokumento d_i, paprasčiausiu atveju apskaičiuojamas kaip euklidinis atstumas tarp vektorių atitinkančių *d* ir d_j; $d_i \in$ *k-NN* – reiškia, kad d_i yra iš *k artimiausių d* kaimynų funkcijos *sim()* atžvilgiu; *y*(d_i,c_j) – funkcija, įgyjanti reikšmę 1 kai dokumentas d_i priklauso kategorijai c_j ir 0 kitu atveju. Apskaičiavus šiuos rodiklius kategorijos priskyrimas vykdomas pagal formulę:

$$\arg\max_{j=1,\ldots,m}(s(d,c_j)),$$

kur $c_1,\ldots,c_m$ – iš anksto apibrėžtos kategorijos.

Egzistuoja kelios algoritmo k-NN atmainos. Kitaip gali būti apskaičiuojama dokumento priskyrimo kategorijai funkcija, atstumas tarp dokumentų. Šaltinyje [ZD02] siūloma pritaikyti *tf*id*f (angl. *term frequency – inverse document frequency*) svorių schemą ir dokumentų panašumą apskaičiuoti vietoj euklidinio atstumo panaudojant kosinusinį panašumą (angl. *cosine similarity*). Tokiu atveju duotiems dokumentams d_1 ir d_2 juos atitinkantys vektoriai su svoriais yra $V_1 = (w_{11},\ldots,w_{1n})$ ir $V_2 = (w_{21},\ldots,w_{2n})$, kur

$$w_{i,j} = \frac{freq_{i,j}}{\max_i(freq_{i,j})} \times \log\frac{N}{n_i},$$

kur $freq_{i,j}$ yra termino t_i pasikartojimo dažnis dokumente d_j; *N* yra bendras dokumentų skaičius ir n_i yra dokumentų, kuriose pasitaiko terminas t_i, skaičius.

Tada panašumo funkcija *sim*(d_1, d_2) yra:

$$sim(d_1,d_2) = \frac{V_1 * V_2}{\|V_1\|_2 \|V_2\|_1} = \frac{\sum_{i=1}^{n} w_{1i} \times w_{2i}}{\sqrt{\sum_{i=1}^{n} w_{1i}^2}\sqrt{\sum_{i=1}^{n} w_{2i}^2}}.$$

Pasinaudosime jau nagrinėtu grožinės/mokslinės literatūros klasifikavimo pavyzdžiu. Tarkime, turime tokius pat pradinius duomenis ir reikia klasifikuoti tokį pat naują objektą k-NN metodu, pasirinkus parametrą *k* = 5. Kadangi skaičiuosime atstumus tarp vektorių, skaitmeniškai užkoduojame savybių galimas įgyti reikšmes:

- „nesutinkama“ = 0;
- „retai“ = 1;
- „dažnai“ = 2.

Paskaičiuojame atstumą tarp ieškomo naujo objekto vektoriaus ir pradinių duomenų vektorių:

$$d_1 = \sqrt{(0-1)^2 + (2-1)^2 + (2-0)^2} = \sqrt{6}$$
$$d_2 = \sqrt{(1-1)^2 + (1-1)^2 + (1-0)^2} = 1$$
$$d_3 = \sqrt{(0-1)^2 + (2-1)^2 + (0-0)^2} = \sqrt{2}$$
$$d_4 = \sqrt{(1-1)^2 + (2-1)^2 + (0-0)^2} = \sqrt{2}$$
$$d_5 = \sqrt{(1-1)^2 + (0-1)^2 + (0-0)^2} = 1$$
$$d_6 = \sqrt{(2-1)^2 + (1-1)^2 + (1-0)^2} = \sqrt{2}$$

Pasirenkame 5 artimiausius klasifikuojamo vektoriaus kaimynus d_2, d_3, d_4, d_5, d_6. Kadangi 2 iš jų atstovauja mokslinės literatūros klasei, o 3 – grožinės, darome išvadą, kad naujas tekstas priklauso būtent jai.

2.3 Atraminio vektoriaus mašinos

Atraminio vektoriaus mašinos (angl. *Support Vector Machines, SVM*) kaip ir k-NN priklauso prižiūrimo mokymo (angl. s*upervised learning*) klasifikavimo metodams. Tai sąlyginai naujas, pasiūlytas 1995 m. algoritmas. Duotai apmokomųjų duomenų aibei kurią sudaro tiesiškai atskiriami vektorinėje erdvėje “teigiami” ir “neigiami” pavyzdžiai, šis algoritmas sukuria hiperplokštumą (angl. *hyperplane*) kuri perskiria duomenis taip, kad atstumas tarp artimiausių pavyzdžių ir hiperplokštumos yra maksimizuotas. Šios plokštumos radimas gali būti realizuotas kvadratinio sudėtingumo algoritmais. SVM gali būti išplėstas ir duomenims kurie negali būti tiesiškai atskirti vektorinėje erdvėje, pritaikant specialius hiperplokštumos apibrėžimo būdus arba padidinant pradinės vektorinės erdvės dimensiją.

Šis metodas gali būti naudojamas ne tik dviejų klasių klasifikavimui, tačiau gali būti išplėstas ir iki *m* klasių. Tuomet reikės *m* hiperplokštumų norint sudalinti vektorinę erdvę dokumentų klasifikavimui.

Šaltinyje [MS01] pateikimas matematinis SVM modelis. Apmokymo duomenys apibrėžiami kaip: $(x_1,y_1),\ldots,(x_l,y_l)$, kur $x_i \in \Re^n$, $y_i \in \{-1,+1\}$, $i \in [1,\ldots,l]$. Klasifikavimo funkcija apibrėžiama:

$$g(x) = sign(f(x))$$
$$f(x) = \sum_{i=1}^{l} y_i \alpha_i K(x_i, x) + b,$$

kur *K* yra branduolio funkcija; $b \in \Re$ ir yra slenkstis; α_l yra svoriai. Be to svoriai tenkina tokias nelygybes:

$$\forall i : 0 \le \alpha_i \le C;$$
$$\sum_{i=1}^{l} \alpha_i y_i = 0,$$

kur *C* yra neteisingo klasifikavimo kaina. Vektoriai x_i su ne nuliniu α_i yra vadinami pagalbiniais vektoriais. Tiesinėms SVM branduolio funkcija *K* apibrėžiama:

$$K(x_i, x) = x_i \cdot x .$$

Tokiu atveju galime perrašyti funkciją *f(x)* taip:

$$f(x) = w \cdot x + b ,$$

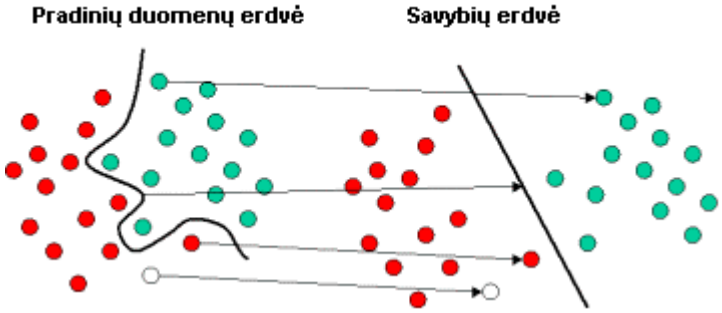
kur

$$w = \sum_{i=1}^{l} y_i \alpha_i x_i .$$

„Apmokyti“ SVM reiškia rasti α_i ir *b* išspendžiant sekančią optimizacijos problemą:

$$\begin{cases} \max(\sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j K(x_i, y_j)); \\ \forall i : 0 \le \alpha_i \le C; \\ \sum_{i=1}^{l} \alpha_i y_i = 0. \end{cases}$$

Sprendimas duoda optimalią hiperplokštumą, kuri yra klasifikavimo pasirinkimo kraštas tarp dviejų klasių.



2.3.1 pav. Pradinių duomenų atskyrimas hiperplokštuma

2.3.1 pav. [NA01] pateikiama iliustracija parodo kaip tiesiškai neatskiriami pradiniai duomenys branduolio funkcijos pagalba yra atskiriami. Tai leidžia vietoj sudėtingos kreivės atskyrimui pasitelkti hiperplokštumą.

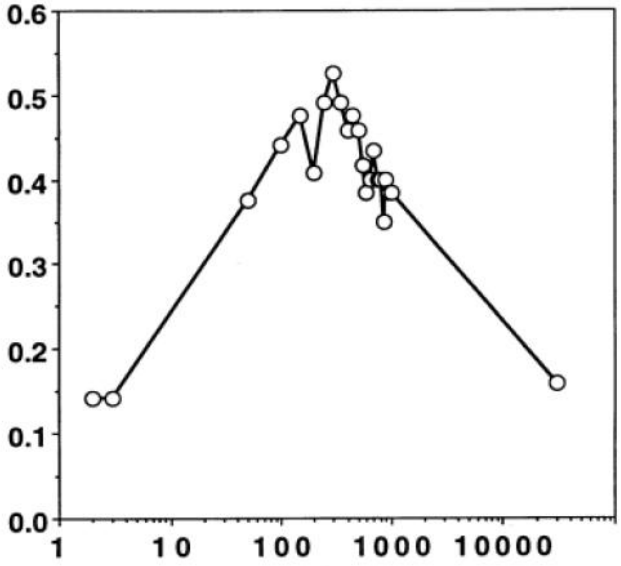
2.4 Paslėptos semantinės analizės (indeksavimo) modelis

Dokumentų paieška remiantis tik užklausoje paminėtais žodžiais gali duoti blogus rezultatus, kadangi sietinuose su užklausa tekstuose gali nebūti paminėti jos užklausos žodžiai. Šaltinyje [AC03] aprašomas paslėptos semantinės analizės (indeksavimo) (angl. *latent semantic analysis, latent semantic indexig, LSA/LSI*) metodas siūlomas

tokios problemos sprendimui. Pagrindinė idėja susieti kiekvieną dokumento ir užklausos vektorių su žemesnės eilės vektorine erdve, kuri savo ruožtu yra susieta su sąvokomis ir vykdyti dokumento paiešką toje erdvėje. Tokia paieška bus efektyvesnė, taipogi reikės mažiau skaičiavimų, kadangi vektorinės erdvės dimensija yra mažesnė. LSA algoritmas vykdomas tokiais etapais:

- Sudaroma tekstų-sąvokų matrica *X*, kurios stulpeliai atitinka pradinius tekstus, o eilutės – raktinius terminus. Matricos reikšmę $x_{i,j}$ sudaro raktinio termino su indeksu *I* pasikartojimų skaičius tekste *j*.
- Naudojant SVD (angl. *singular value decomposition*) metodą sudaroma sandauga *X* = *T* * *S* * *D*, tokią kad *T* ir *D* matricos turi ortonormuotus stulpelius ir yra atitinkamai kairio ir dešinio singuliarumo vektorių matricos, o *S* yra diagonalinė singuliarumo reikšmių matrica.
- Jei *S* reikšmės yra surūšiuotos pagal dydį, pirmos didžiausios *k* reikšmės yra paliekamos, o kitos prilydinamos nuliui.
- Tada atliekama sandauga *X* = *T* * *S* * *D*, su pakeista matrica *S*. Gauta matrica *Y* yra apytiksliai lygi *X* ir yra *k*-tosios eilės.
- Naudojantis matrica *Y* yra skaičiuojami atatumai tarp terminų vektorių (naudojantis euklidiniu, kosinusiniu ar kitu atstumo matu).
- Randami terminai su nedideliu tarpusavio atstumu (kuo mažesnis atstumas tarp terminų vektorių, tuo jie labiau susiję).
- Paiešką galima vykdyti ne tik pagal terminus sutinkamus užklausoje, tačiau ir pagal rastus susijusius terminus.

Išlieka atviru klausimas, kaip reikia pasirinkti parametrą *k*, kuris nusako gaunamos erdvės dimensiją. Idealiu atveju norima, kad *k* reikšmė būtų pakankamai didelė, kad apimtų visas realias semantines struktūras duomenyse, tačiau pakankamai maža, kad nebūtų duomenų pertekliaus. Šaltinyje [SD01] iliustruojamas sinonimų paieškos testo korektiškų rezultatų koeficientų (Y ašis) ir dimensijų kiekio LSA algoritme (X ašis) priklausomybės grafikas log skalėje. Matome, kad optimalūs rezultatai buvo gauti koeficientui *k* įgijus reikšmę ~300.



2.4.1 pav. Sinonimų paieškos testo korektiškų rezultatų priklausomybė nuo LSA algoritmo koeficiento *k*.

2.5 Hierarchinis klasifikavimas

Hierarchiniame klasifikavime (angl. *hierarchical classification*) objektai yra priskiriami klasėms, kurios yra atvaizduojamos hierarchinėje struktūroje. Dideliam klasių skaičiui toks sprendimas palengvina klasifikavimo rezultatų peržiūrą ir paiešką. Taipogi hierarchinis klasifikavimas leidžia naudotis principu “skaldyk ir valdyk”, atliekant klasifikavimo užduoties dekompoziciją į aibę mažesnių užduočių, kurios vykdomos skirtinguose semantinės struktūros lygiuose. Čia klasifikavimas gali būti vykdomas naudojantis “plokščiais” (angl. *flat*) metodais, tokiais kaip Bayes klasifikatorius, SVM ir pan. Toks būdas sumažina užduoties vykdymo laiką, kadangi kiekviename klasifikavimo etape reikia operuoti mažesniu galimu priskyrimo klasių skaičiumi.

Daugumoje hierarchinio klasifikavimo metodų naudojamos medžio pavidalo struktūros. Apibendrinant galima išskirti keturis jų tipus:

- virtualus kategorijų medis (angl. v*irtual category tree*) – struktūra, kurioje klasės atvaizduojamos medžio viršūnėse. Klasė gali turėti tik vieną tėvą, o dokumentai gali būti priskiriami tik medžio lapuose esančioms klasėms;
- kategorijų medis (angl. *category tree*) – virtualaus kategorijų medžio praplėtimas, leidžiantis priskirti dokumentus visoms jame esančioms klasėms (esančioms ir medžio lapuose, ir vidinėse viršūnėse);
- virtualus kryptinis aciklinis kategorijų grafas (angl. *virtual directed acyclic category graph*) – struktūros pagrindą sudaro kryptinis aciklinis grafas (angl.

directed acyclic graph, DAG). Dokumentai gali būti priskiriami tik klasėms, esančioms išorinėse grafo viršūnėse;

- kryptinis aciklinis kategorijų grafas (angl. *directed acyclic category graph*) – struktūros pagrindą sudaro kryptinis aciklinis grafas, dokumentai gali būti priskiriami bet kokioje viršūnėje esančioms klasėms.

Hierarchinio klasifikavimo metodai tarpusavyje skiriasi ir pasirinktos hierarchinės struktūros "apėjimo" strategijomis. Išskiriami du atvejai:

- didžiojo sprogimo (angl. *big-bang*) – klasifikavimui pateikiamos visos pagal pasirinktą struktūrą tinkamos klasės;
- nusileidimo iš viršaus žemyn (angl. *top-down*) – lygiais paremtas metodas, kai klasifikuojant leidžiamasi struktūra iš viršaus žemyn, kiekviename lygyje naudojant atskirą "plokščią" klasifikatorių.

Didžiojo sprogimo, skirtingai nei nusileidimo metodas, gali naudoti kategorijų struktūros informaciją tik apmokymo, bet ne klasifikavimo fazėje. Taip pat jam gali trūkti lankstumo atsirandant naujoms kategorijoms, t.y. gali būti reikalingas naujas algoritmo apmokymas. Didžiausiu nusileidimo metodo trūkumu galima įvardinti problemą iškylančia neteisingai priskyrus dokumentą vienai iš tėvinių klasių. Tada klasifikavimas gali "nusileisti" žemyn ne ta medžio šaka, kas gali dar labiau iškreipti rezultatus. Taipogi nusileidimo metodui reikia daugiau mokymo duomenų, kadangi reikia apmokyti kiekvieno lygio klasifikatorius.

Šaltinyje [AS01] be jau pateiktų hierarchinio klasifikavimo apibrėžimų nagrinėjami šio klasifikavimo būdo efektyvumo matai (angl. *performance measure*) lyginant su "plokščio" klasifikavimo matais. Teigiama, kad matai, naudojami "plokščių" klasifikatorių atveju nevisiškai atspindi hierarchinio klasifikavimo poreikius. Taip yra todėl, kad klasifikatoriaus priimti sprendimai hierarchinės struktūros viršūnėse yra neįskaičiuojami į klasifikavimo mato apskaičiavimą. Autorių pasiūlymo esmė yra vietoj įprastinių matų skaičiuojančių teisingo/neteisingo klasifikavimo procentus (ar matų papildomai atsižvelgiančių į kai kurių klasių panašumą), naudoti matus papildomai naudojančius atstumą tarp klasių (angl. *category distance*). Atstumas tarp klasių apibrėžiamas kaip jas jungiančių medžio ryšių skaičius.

Eksperimentai buvo atliekami su kategorijų medžio struktūra, naudojantis nusileidimo metodu. Klasifikavimo vykdymui tokioje struktūroje reikalingi du klasifikatorių tipai (vėliau šis klasifikavimo metodas bus įvardinamas kaip STD):

- vietinis klasifikatorius (angl. *local classifier*) – susietas su medžio viršūne klasifikatorius, nustatantis, ar dokumentas priklauso šiai klasei;

- pomedžio klasifikatorius (angl. *subtree classifier*) - susietas su medžio viršūne klasifikatorius, nustatantis, ar dokumentas gali būti siejamas su viršūnės vaikų klasėmis.

Modelyje buvo realizuoti SVM klasifikatoriai. Testams bei apmokymui panaudoti Reuters-21578 duomenys. Eksperimentai parodė, jog siūlomas klasifikavimo rezultatų vertinimo matas pakankamai gerai atspindėjo klasifikavimo rezultatus visais atvejais, ir labai gerai klasėms, kuriose apmokymo dokumentų skaičius buvo didelis (> 200). Detalius eksperimentų duomenis bei matematinį pagrindimą žr. [AS01].

Šaltinyje [AS04] aptariama kita hierarchinio klasifikavimo problema, kuri įvardinama kaip blokavimas (angl. *blocking*). Jos esmė – aukštesnių medžio lygių klasifikatorių neteisingas dokumento atmetimas, taip nesuteikiant jiems galimybę būti klasifikuojamiems žemesniuose lygiuose. Autoriai įveda blokavimo faktoriaus (angl. *blocking factor*) matą, kuris yra apskaičiuojamas medžio viršūnei, kaip proporcija jos blokuojamų dokumentų, priklausančių tos viršūnės vaikų kategorijoms. Pagal šį matą buvo vertinami hierarchinio klasifikavimo metodai:

- STD – jau aprašytas standartinis hierarchinio klasifikavimo metodas;
- Slenksčio mažinimo metodas (angl. *threshold reduction method, TRM*) – kiekvienam medžio lygiui aprašomas slenkstis, kuris įtakoja sprendimą perduoti dokumentą į žemesnį lygį (kuo mažesnis slenkstis, tuo lengviau dokumentas perdudamas). Kiekvienam žemesniam lygiui slenkstis yra mažinamas, taigi didėja galimybė dokumentui leistis medžiu žemyn ir mažėja blokavimo rizika. Metodo trūkumas – žemesnių lygių būtinybė vykdyti tikslesnį klasifikavimą, kadangi didėja neteisingo klasifikavimo galimybė;
- Apriboto balsavimo metodas (angl. *restricted voting method, RVM*) – suteikia galimybę aukštesnio lygio klasifikatorių palikuonims klasifikuoti dokumentą lygiagrečiai su jais. Tada sprendimas ar leisti dokumentą žemyn medžiu yra priimamas kelių tarpusavyje nepriklausomų klasifikatorių balsavimu;
- Praplėstas daugybos metodas (angl. *extended multiplicative method, EMM*) – kaip ir RVM naudoja slenkstį kiekviename medžio lygyje. Jei tikimybės kad dokumentas priklauso *n*-tojo lygio klasei ir tikimybės, kad dokumentas priklauso duotosios klasės tėvinei klasei, sandauga didesnė arba lygi *n*-tojo lygio slenksčiui, tai dokumentas priskiriamas *n*-tojo lygio klasei.

Atliktuose eksperimentuose autoriai implementavo SVM klasifikatorius ir naudojo virtualių kategorijų medžius kaip hierarchines struktūras. Duomenų šaltiniu buvo pasirinkta jau minėta Reuters-21578 kolekcija. Rezultate įvertinus kiekvieno iš metodų

blokavimo faktorius jų efektyvumas pagal šį matą nustatytas taip: RVM >TRM > EMM > STD (kur > žymi geresnį metodą). Straipsnyje [AS04] autoriai pateikia detalius eksperimentų duomenis ir apibrėžimus.

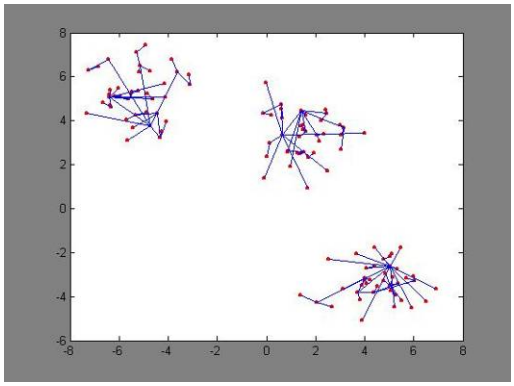
Šaltinyje [MS98] efektyviam Web dokumentų hierarchiniam klasifikavimui siūloma pritaikyti paremtą taisyklėmis (angl. *rule based*) klasifikavimo metodą RIPPER (angl. *Repeated Incremental Pruning to Produce Error Reduction*). Metodo esmę sudaro raktinių terminų susiejimas paprastomis loginėmis operacijomis į taisyklių rinkinį, kuris savo ruožtu yra susiejamas su klase. Taigi dokumentai atitinkantys klasės taisyklių rinkinį yra su jai priskiriami. Autoriai parodo, kad RIPPER hierarchinio klasifikavimo variantas yra kur kas efektyvesnis už jo "plokščią" variantą.

2.6 Grupavimo metodai

Grupavimo (angl. *Clustering*) metodų tikslas – sumažinti klasifikavimui reikalingus duomenų ir skaičiavimų kiekius, grupuojant panašias savybes turinčius duomenis. Gera šių metodų panaudojimo galimybė yra automatinis klasifikatorių ar taksonomijų generavimas, kadangi grupavimas vykdomas pagal žmogaus mąstymui būdingas taisykles, taigi algoritmų darbo rezultatas irgi yra žmogui intuityviai priimtinas. Taipogi automatizuojant klasifikavimo procesus minimizuojamas „žmogiškasis faktorius".

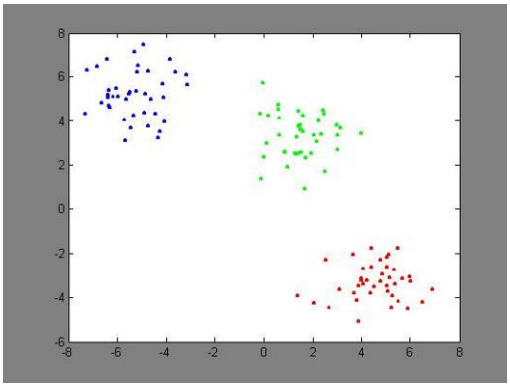
Pagal [SK97], grupavimo metodai gali būti suskirstyti į du paprastus tipus: hierarchinius (angl. *hierarchical*) ir padalinimo (angl. *partitional*). Kiekvienam iš tipų egzistuoja nemažai potipių bei skirtingų algoritmų, nusakančių grupių radimą.

Hierarchinis grupavimas vyksta nuosekliai, arba jungiant mažesnes grupes į didesnes, arba dalinant didesnes grupes. Algoritmų skirtumas pasireiškia nusakant taisyklę, pagal kurią sprendžiama kurias grupes reikia apjungti ar skaidyti. Rezultate gaunamas grupių medis vadinamas dendograma (angl. *dendogram*), kuris parodo kaip grupės yra tarpusavyje susijusios. Hierarchiškai sugrupuoti objektai pavaizduoti 2.6.1 pav. schemoje (šaltinis [NA00].)



2.6.1 pav. Hierarchiškai sugrupuoti objektai

Padalinimo grupavimo metodai bando tiesiogiai skaidyti į aibę tarpusavyje nesusijusių grupių, pagal tam tikrus nustatytus parametrus (2.6.2 pav., [NA00]).



2.6.2 pav. Padalinimu sugrupuoti objektai

Kriterijaus funkcija kurią bando minimizuoti padalinimo algoritmas gali pabrėžti lokalią duomenų struktūrą, priskiriant grupes tikimybinio pasiskirstymo funkcijos pikams ar globaliai struktūrai. Paprastai globalūs kriterijai įtraukia mato skirtumų minimizavimą grupėje sutinkamiems pavyzdžiams, maksimizuojant mato skirtumus skirtingoms grupėms.

Taipogi šaltinyje [SK97] pateikiamas dažnai sutinkamas padalinimo grupavimo metodo pavyzdys K-vidurkių grupavimas (angl. *K-mean clustering*). Šiame metode kriterijaus funkcija yra vidutinis kvadratinis duomenų elementų x_k atstumas nuo jų artimiausių grupės centroidų:

$$E_K = \sum_{i=1}^{K} \left\| x_k - m_{c(x_k)} \right\|^2 ,$$

kur $c(x_k)$ yra indeksas centroido kuris yra artimiausias x_k. Vienas galimas minimizuojantis kainos funkciją algoritmas pradeda nuo inicializavimo K grupių centroidų, pažymėtų m_i, kur $i=1,\ldots,K$. Tada m_i pozicijos yra iteratyviai pataisomos pirma priskiriant duomenų pavyzdžius prie artimiausių grupių ir tada perskaičiuojant centroidus. Iteracijos nebeskaičiuojamos, kai E nustoja ženkliai keistis.

Grupavimo metodų trūkumu galima įvardinti sudėtingumą susijusį su grupių interpretavimu. Dauguma grupavimo algoritmų teikią pirmenybę tam tikroms formoms, ir duomenys juose visada bus priskiriami toms formoms, net jei duomenyse nėra grupių. Taigi grupavimo analizės rezultatai turi būti peržiūrimi ir patvirtinami.

Kita potenciali problema yra tai, kad grupių kiekio pasirinkimas gali turėti lemiamos įtakos: skirtingos grupės gali atsirasti kintant K. Geras centroidų inicializavimas taipogi turi didelę reikšmę, jų blogas pasirinkimas gali palikti kai kurias grupes tuščiomis.

Grupavimas metodų optimizavimas aptariamas šaltinyje [ID02]. parodomas tokio metodo efektyvumas. Autoriai pasiūlė grupavimo algoritmą, minimizuojantį

pasiskirstymą tarp vienai grupei priklausančių objektų ir maksimizuojantį atstumą tarp įvairių grupių. Savo eksperimentams jie panaudojo dvidešimties naujienų grupių žinutes (viso 20000) ir trijų lygių virtualų kategorijų medį su 49 lapais bei jam jau prikirtais 5000 dokumentų, implementavo Bayes ir SVM klasifikatorius. Buvo parodyta, kad pasiūlytas padalinimo algoritmas atlieka efektyvesnį žodžių grupavimą nei kiti anksčiau pasiūlyti metodai, kas ženkliai padidino klasifikavimo tikslumą. Detalus teorinis pagrindimas ir eksperimentų rezultatai pateikiami [ID02].

2.7 Kiti klasifikavimo metodai

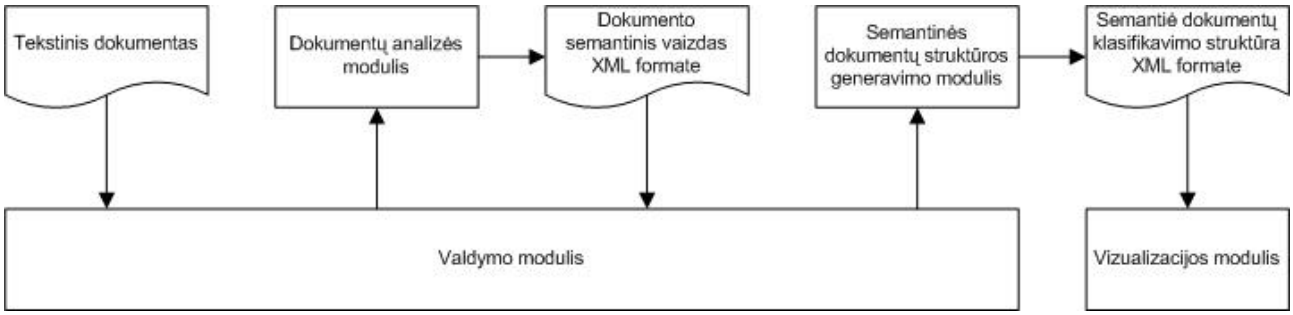
Egzistuoja nemažai čia neapžvelgtų klasifikavimo metodų. Tai ir populiaresni ir geriau ištyrinėti, tokie kaip, centrinio vektoriaus algoritmas (angl. *centroid vector*, aprašomas [TT02]), tiesinis mažiausių stačiakampių metodas (angl. *linear last squares fit*), neuroniniai tinklai (angl. *neural networks*), sprendimų medžiai (angl. *decissin trees*), ir mažiau žinomi, atskirų autorių minimi metodai, tokie kaip SRFW (angl. *Simple, Fast and Effective Text Classification Algorithm,* aprašomas [ZD02]). Taipogi egzistuoja didelis skaičius algoritmų modifikacijų, vienaip ar kitaip pagerinančių jų darbą. Visgi esminiai jau aukščiau minėti klasifikavimo principai išlieka jiems būdingi.

3 Programų sistemos projektas

Vienas iš šio darbo tikslų – sukurti realiai veikiančią programinės įrangos sistemą, leidžiančią automatiškai klasifikuoti lietuviškus tekstus ir pateikti vartotojo sąsają dokumentų klasifikavimo semantinės struktūros peržiūrai bei dokumentų paieškai. Prioritetinėmis programų sistemai išskiriamos šios savybės:

- semantinės dokumentų klasifikavimo struktūros vizualizacijos paprastumas ir prieinamumas vartotojui;
- galimybė keisti semantinės dokumentų analizės algoritmus;
- galimybė keisti semantinę dokumentų klasifikavimo struktūrą.

Siekiant įgyvendinti nurodytus tikslus, uždavinio sprendimas skaidomas į atskirus, nepriklausomus modulius, kurie tarpusavyje keičiasi XML formato bylomis. Programų sistemos schema pateikiama 3.1 pav. Rodyklėmis joje pažymėtas principinis duomenų apdorojimo kelias.



3.1 pav. Dokumentų klasifikavimo programinės įrangos sistemos schema.

Modulių implementuojamas funkcionalumas bei realizacijos būdas:

- Valdymo modulis – suteikia vartotojo sąsają klasifikuojamų dokumentų nustatymui, dokumentų analizės modulio pasirinkimui, semantinės duomenų struktūros generavimo modulio pasirinkimui. Taipogi jis iškviečia nurodytus modulius su vartotojo pasirinktais parametrais. Realizuojamas kaip Windows aplikacija.
- Dokumentų analizės modulis – analizuoja valdymo modulio nurodytus dokumentus, sudaro jų semantinio vaizdo dokumentus XML formate. Realizuojamas kaip komandinės eilutės (angl. *Command Line*) aplikacija, be vartotojo sąsajos.
- Semantinės dokumentų klasifikavimo struktūros generavimo modulis – analizuoja valdymo modulio nurodytus dokumentų semantinius vaizdus,

pagal gautus rezultatus keičia semantinę klasifikavimo struktūrą. Realizuojamas kaip komandinės eilutės aplikacija, be vartotojo sąsajos.

- Vizualizacijos modulis – suteikia vartotojo sąsają semantinės dokumentų klasifikavimo struktūros peržiūrai, dokumentų paieškai. Realizuojamas kaip Web aplikacija.

Taip suprojektuota programų sistema leidžia nepriklausomą kiekvieno iš modulių implementaciją, atsižvelgiant tik į generuojamų XML bylų formatus. Taipogi galimas dokumentų analizės modulio bei semantinės dokumentų klasifikavimo struktūros generavimo modulio realizavimas vienu moduliu.

4 Kalbos apibrėžimai ir žodžių analizės algoritmai

Kadangi iškeltas darbo tikslas numato lietuviškų tekstų analizę, būtina pateikti tam tikrus kalbos apibrėžimus, kurie leistų aiškiau suvokti darbo tikslą. Taipogi reikalingi instrumentai atlikti analizę ne tik tekstų, tačiau ir žodžio lygmenyje.

4.1 Kalbos apibrėžimai

Automatinio klasifikavimo uždavinys numato, kad nėra iš anksto apibrėžtų kategorijų, kurioms galima būtų priskirti tekstus. Todėl būtina tas kategorijas išskirti iš pačių dokumentų. Taigi vieni ar kiti teksto žodžiai (ar jų formos, dalys) taps kategorijomis. Nustatyti taisykles kaip iš žodžių iškirti kategorijas padės morfologijos apibrėžimai (pagal [AU06]):

- **Žodis** – teksto dalis, suvokiama kaip sąlygiškai atskira nuo to paties lygmens kitos dalies ir rašoma atskirai.
- **Šaknis** - yra pagrindinė žodžių dalis, prisijungianti afiksus ir galinti susijungti su kita šaknimi. Afiksai – tarptautinis bendras priešdėlių, priesagų, galūnių ir kitų žodžio dalių, išskyrus šaknį, pavadinimas. Kadangi ta pati **šaknis** prisijungia įvairias žodžių ir jų formų sudedamąsias dalis, ji apibrėžiama ir kaip **bendroji giminiškų žodžių dalis.**
- **Kamienas** – žodžio dalis be galūnės.
- **Sąvoka** – tai susidarytas apie kokį nors daiktą, reiškinį ar veiksnį supratimas. Galima teigti, kad sąvoka yra labai artimas terminas kategorijai, tolimesniame darbe laikysime jog tai sinonimai.

Kadangi tekstuose sutinkami žodžiai, o turime išskirti sąvokas, būtina panagrinėti morfologinį šių terminų santykį.

Sąvoka, kaip tam tikra mąstymo forma, atspindi esminius daikto ar reiškinio požymius, apibendrina žmogaus patirties bei pažinimo duomenis. Plečiantis pažinimui, didėja sąvokos apimtis, tikslėja jos turinys, atsiranda naujų sąvokų. Nors sąvokos reiškiamos žodžiais, bet ne visi žodžiai reiškia sąvokas. Nereiškia sąvokų vad. tarnybiniai žodžiai (jungtukai, prielinksniai, dalelytės), jaustukai, kurie yra tiktai tam tikri emociniai signalai, taip pat įvardžiai, kurie arba pavaduoja savarankiškas kalbos dalis, arba atlieka nurodomąją funkciją. Nereiškia sąvokų ir tikriniai žodžiai (*Petras, Alytus, Nemunas*).

Kitas sąvokos ir žodžio skirtumas tas, kad žodžiai neretai sudaromi vieno kurio, kartais ir neesminio, daikto ar reiškinio požymio pagrindu, o sąvokos – tiktai esminių

požymių apibendrinimo pagrindu. Dėl to sąvoka yra bendražmogiška mąstymo kategorija, o žodis turi nacionalinę specifiką (ta pati sąvoka atskirose kalbose dažniausiai reiškiama skirtingais žodžiais, plg. liet. *miškas;* angl. *forest;* vok. w*ald;* ir kt.). Skiriasi sąvoka nuo žodžio dar tuo, kad ji gali būti reiškiama keliais žodžiais, sintaksiniais žodžių junginiais. Antra vertus, vienas žodis gali reikšti kelias sąvokas. Nors sąvoka ir žodis yra skirtingi dalykai (sąvoka – mąstymo, logikos kategorija, o žodis – kalbos kategorija), vis dėlto tarp jų egzistuoja glaudi tarpusavio sąsaja. Iš vienos pusės, žodis yra sąvokos formavimo priemonė, o iš antros – sąvoka įsikūnija žodyje.

Praktinį žodžio ir sąvokos santykį nustatysime nagrinėdami duomenų struktūras, kurias naudosime dokumentų semantiniam vaizdui aprašyti, bei algoritmus tekstų analizei.

4.2 Žodžio šaknies nustatymo algoritmas

Kiekvieną lietuvišką žodį sudaro tokios žodžio dalys: priešdėliai, šaknis, priesagos, galūnės. Nauji žodžiai padaromi prie šaknies pridedant priesagas, priešdėlius, keičiant galūnes ir suduriant. Žodžio šaknies nustatymo algoritmas remiasi principu, jog visos žodžio dalys išskyrus šaknį yra žinomos, ir gali būti nustatytos iki žodžio analizės. Tuomet paėmę žodį ir nuosekliai atmetę žinomas žodžio dalis gauname žodžio šaknį. Lentelėje pateikiamos žodžio dalys kurios buvo naudojamos analizėje kaip pradiniai duomenys (remiantis [NS89]):

Priešdėliai	Priesagos	Galūnės	Priešdėliai	Priesagos	Galūnės
ANT	AD	A	UŽUO	INT	IAUS
ANTI	AL	Ą		IOM	IEMS
API	AM	AI		IOT	IES
APY	AN	AIS		IŲ	IO
AT	AMS	ANTIS		IUOJ	IOS
ATA	ANTI	AS		IUOT	IS
ATI	ANT	ĄS		IUOS	ĮS
ATO	ATOR	ASIS		YB	IU
BE	AUJ	AU		YKL	IŲ
Į	AUT	AUS		YM	IUS
INFRA	AV	E		YN	Y
IŠ	ČIOM	Ę		YT	YJE
NE	DAV	Ė		YS	YS
NU	DĖT	EI		JAM	O
NUO	DYT	EIS		OJ	OME
PA	EIV	ĖJE		OK	OMS
PAR	ĖJ	ĖMS		OM	OS
PER	ĖL	ENS		OS	OSE
PO	ĖM	ERS		OT	S
PRA	EN	ES		OV	U
PRI	ENT	ĘS		SĖT	Ų
PRIE	ĖT	ĖS		SK	UI

PRIEŠ	IA	ĖSE		SOT	ŲJŲ
PRO	IAM	ĘSTI		STYT	UO
SĄ	IAU	I		ŠĖT	US
SAM	IAV	Į		ŠOT	ŲS
SAN	IEJ	IA		TELĖT	ŪS
SI	IJ	IĄ		TI	USI
SU	IM	IAI		UOJ	
TE	IN	IAIS		UOJAM	
UŽ	INĖT	IAMS		UOT	
UŽU	INK	IAS		UOSE	

Žemiau pateikiamas šaknies gavimo algoritmas:

- Atmesti žodžio galūnę
    - Nustatyti *n* = 4
    - Tikrinti, ar žodžio paskutiniai n simbolių nėra viena iš galūnių sąrašo reikšmių
        - Jei galūnė atpažinta, atmesti žodžio *n* paskutinių simbolių ir grąžinti žodį tolimesnei analizei
        - Jei galūnė neatpažinta, sumažinti *n* (*n* = *n* – 1) ir tikrinti iš naujo
    - Galūnė nerasta, grąžinti žodį tolimesnei analizei
- Atmesti žodžio priesaga
    - Nustatyti *n* = 5
    - Tikrinti, ar žodžio paskutiniai n simbolių nėra viena iš priesagų sąrašo reikšmių
        - Jei priesaga atpažinta, atmesti žodžio n paskutinių simbolių ir grąžinti žodį tolimesnei analizei
        - Jei priesaga neatpažinta, sumažinti *n* (*n* = *n* – 1) ir tikrinti iš naujo
    - Priesaga nerasta, grąžinti žodį tolimesnei analizei
- Jei priesaga buvo rasta, bandyti atmesti priesagą pakartotinai
- Atmesti žodžio priešdėlį
    - Nustatyti *n* = 5
    - Tikrinti, ar žodžio pirmieji *n* simbolių nėra viena iš priešdėlių sąrašo reikšmių
        - Jei priešdėlis atpažintas, atmesti žodžio n pirmųjų simbolių ir grąžinti žodį tolimesnei analizei
        - Jei priešdėlis atpažintas, sumažinti *n* (*n* = *n* – 1) ir tikrinti iš naujo

    - Priešdėlis nerastas, grąžinti žodį tolimesnei analizei
- Jei priešdėlis buvo rastas, bandyti atmesti priešdėlį pakartotinai
- Likusi žodžio dalis yra šaknis

Toks žodžio analizės algoritmas užtikrina pakankamai aukštą šaknies radimo tikimybę. Neteisingai žodžio šaknis gali būti nustatoma trūkstant pradinių duomenų, sudurtiniams žodžiams, pavadinimams, užsienio kalbų žodžiams ir pan. Tačiau šio darbo kontekste užtikrinti šimtaprocentinį šaknies radimo tikslumą nebūtina, dėl tokių priežasčių:

- Galutinis sistemos vartotojas susiduria tik su šaknies radimo padariniais, o ne su pačia šaknimi
- Dažniausiai pasitaiko pakankamai paprasti žodžiai, kuriems šaknis randama teisingai
- Jei algoritmas suklydo nustatinėdamas šaknį, tai tokiam pat, ar panašiam žodžiui bus padaryta tokia pat klaida ir jų šaknys sutaps

Žemiau pateikiami algoritmo darbo rezultatai kai kuriems žodžiams, sutiktiems analizuojant pasirinktus tekstus:

Žodis	Šaknis	Žodis	Šaknis	Žodis	Šaknis
transportas	transport-	įstatymas	stat-	geležinkeliečiai	geležinkel-
transportavimas	transport-	poįstatyminis	stat-	vežimas	vež-
transporto	transport-	nuostatai	stat-	pervežinėjimas	vež-
besitransportuojantys	transport-	pastatas	stat-	bevariklėms	varikl-

4.3 Žodžio vienaskaitos vardininko nustatymo algoritmas

Norint sukurti patogią vartotojo sąsają kur bus atvaizduojamos dokumentus apibūdinančios sąvokos, būtina nustatyti ir žodžio vienaskaitos vardininką. Tai galima padaryti pagal esamą žodžio galūnę parinkus vienaskaitos vardininko galūnę iš sąrašo. Tačiau šis uždavinys yra sunkiai išsprendžiamas, kadangi neįmanoma nustatyti vienareikšmiško galūnių ryšio, ką iliustruoja žemiau pateikiama lentelė:

Žodis	Galūnė	Vienaskaitos vardininkas	Galūnė
autobusas (kas?)	-as	autobusas	-as
autobusams (kam?)	-s	autobusas	-as
lengvatoms (kam?)	-s	lengvata	-a
durys (kas?)	-ys	durys	-ys
duris (ką?)	-is	durys	-ys
pažintys (kas?)	-ys	pažintis	-is

Siekiant išvengti pagal leksiką neteisingų žodžių sudarymo vienaskaitos vardininke (pvz. „duris“, „lengvatas“), naudojamas panašių žodžių tarpusavio palyginimo metodas.

Galūnėms, esančioms aukščiau nurodytame sąraše (žr. 4.1), suteikiame atitinkamus koeficientus *k*:

- jei galūnė netinka bet kokios giminės vienaskaitos vardininkui, *k* = 4 (pvz.: -e; -ą; -ai; -ui;)
- jei galūnė gali tikti vyriškos giminės vienaskaitos vardininkui, *k* = 3 (pvz.: -as; -is; -s;)
- jei galūnė gali tikti moteriškos giminės daugiskaitos vardininkui, *k* = 2 (pvz.: -os; -ys;)
- jei galūnė gali tikti vyriškos giminės vienaskaitos vardininkui, *k* = 1 (pvz.: -a; -ė; -i;)

Tuomet lygindami du žodžius, kurių skiriasi tik galūnė, nustatome kuriam iš jų galūnės koeficientas yra mažesnis ir darome prielaidą jog šis žodis yra kandidatas į vienaskaitos vardininkus. Esant pakankamam žodžių kiekiui, ir jose būnant bent vienam vienaskaitos vardininkui, galime su didele tikimybe jį nustatyti:

Pirmas žodis	**Antras žodis**	**Kandidatas į vienaskaitos vardininkus**
autobusas	autobusui	autobusas
lengvata	lengvatos	lengvata
durys	duryse	durys
durys	duris	durys
pažintys	pažintis	pažintys

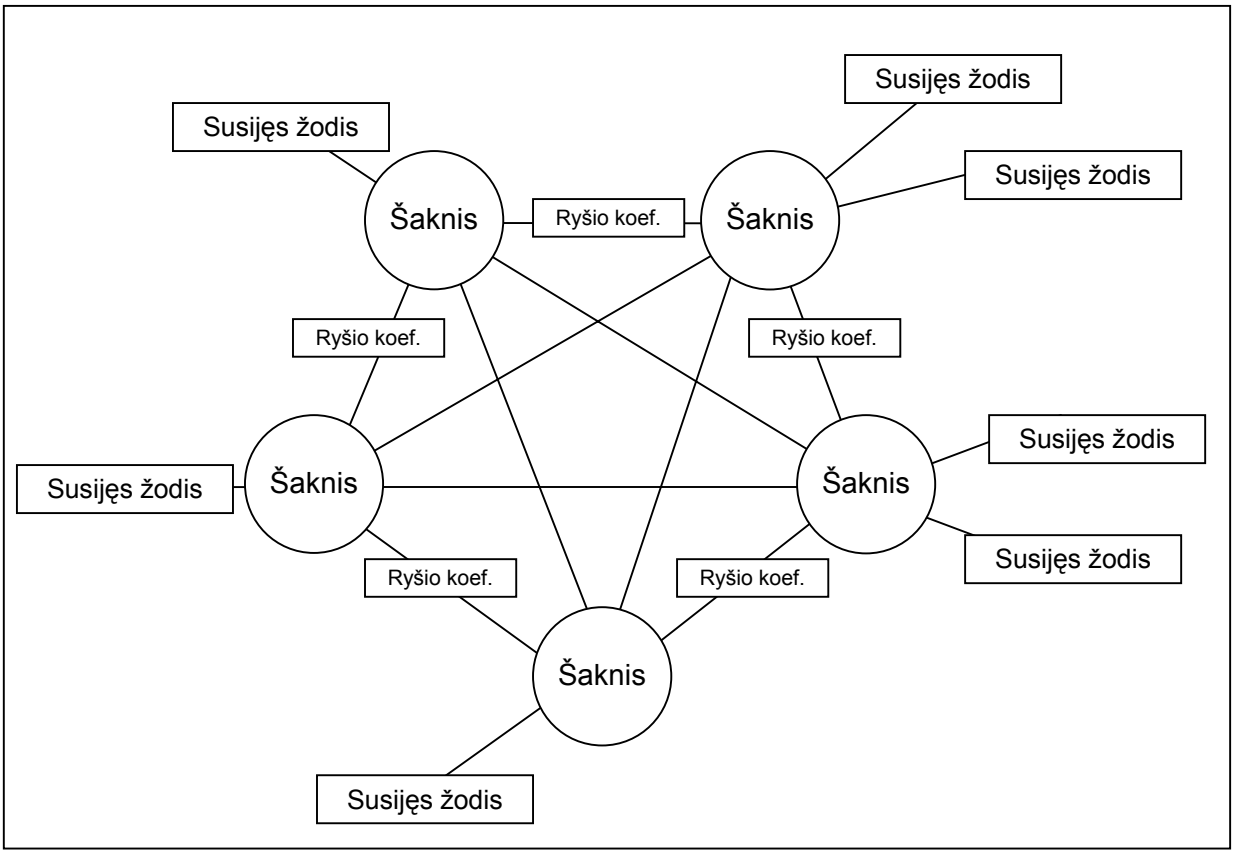
Kaip matome, naudojantis ir šiuo metodu ne visada pavyksta teisingai įvertinti žodžius („pažintys“ – „pažintis“), tačiau daugeliu atveju metodas duos teisingą rezultatą. Taipogi išvengiama dirbtinio žodžio sudarymo, nes kandidatais į vienaskaitos vardininkus tampa tik tekstuose sutikti žodžiai.

5 Duomenų struktūros tekstų apibūdinimui

Darbo tikslas numato pilnai automatinį dokumentų klasifikavimą, todėl duomenų struktūra aprašanti dokumentus negali būti labai sudėtinga, nes sunku išgauti daug semantinės informacijos remiantis statistiniais metodais (kadangi kitokių metodų pritaikyti iš esmės negalima dėl uždavinio sąlygos). Todėl tolimesniam darbui pasirenkama pakankamai paprasta grafo su svoriais duomenų struktūra. Žemiau nagrinėjami keli galimi grafo variantai. (Pastaba. Šiame skyriuje nagrinėsime tik pačių duomenų struktūrų tinkamumą semantiškai aprašyti dokumentams, o metodus kaip jos yra sudaromos praktikoje (nustatomos sąvokos ir ryšiai tarp jų) aptarsime 6.2 ir 6.3 skyriuose).

5.1 Grafas su žodžio šaknimi viršūnėse

Pagal šaknies apibrėžimą, ji yra bendroji giminiškų žodžių dalis. Todėl galime daryti prielaidą, jog tos pačios šaknies žodžiai atstovauja tai pačiai sąvokai (kategorijai). Pavyzdžiui žodžių „transportas“, „transportavimas“, „transportuotojas“ bendra šaknis „transport-“ juos apjungia į vieną sąvoka. Taigi sutapatiname sąvoką su šaknimi. Tokio grafo schema pavaizduota 5.1.1 pav.



5.1.1 pav. Grafo, kurio viršūnėse žodžio šaknys, schema.

Matome, kad grafo viršūnėse saugoma šaknies reikšmė, o žodžiai su tokia šaknimi gali būti traktuojami kaip viršūnės savybės. Lankai tarp viršūnių gali turėti svorius, kurie nusako sąvokų tarpusavio ryšio stiprumą.

Taip galima būtų pateikti vieno dokumento semantinį vaizdą. Apjunginėjant kelių dokumentų semantinius vaizdus į vieną grafą, reikėtų su viršūnėmis susieti dar ir atitinkamus dokumentus.

5.2 Grafas su kamienais viršūnėse

Grafas su žodžių šaknimis viršūnėse yra geras informacijos apibendrinimo prasme, kadangi daugelį žodžių galima susieti su nedideliu skaičiumi viršūnių, atitinkančių sąvokas. Tačiau pasirodo, jog prielaida, kad žodžio prasmę nusako jo šaknis, pakankamai dažnai gali būti neteisinga. Pvz. žodžių „įstatymas“ ir „pastatas“ šaknis ta pati, „stat-“, tačiau prasmė visiškai skirtinga. Tokių pavyzdžių galima sugalvoti nemažai. Todėl norint tiksliai nusakyti sąvoką grafe, galima jo viršūnėse saugoti ne šaknį, o žodžio kamieną („įstat-“, „pastat-“). Tačiau tokiu atveju galime susidurti su problema, jog panašios pagal prasmę sąvokos („įstatymas“, „įstatai“, „nuostatai“) atsidurs skirtingose grafo dalyse ir tarp jų nepavyks nustatyti pakankamai stipraus semantinio ryšio. Taipogi tokio grafo viršūnių kiekis artės prie žodžių kiekio, kas reiškia jog grafe bus mažiau apibendrinančių sąvokų (kaip žinome sąvoka ≠ žodis). Tai blogai ir Informacijos paieškos prasme, nes reikia atlikinėti daugiau palyginimų.

5.3 Kombinuotas šaknų – kamienų grafas

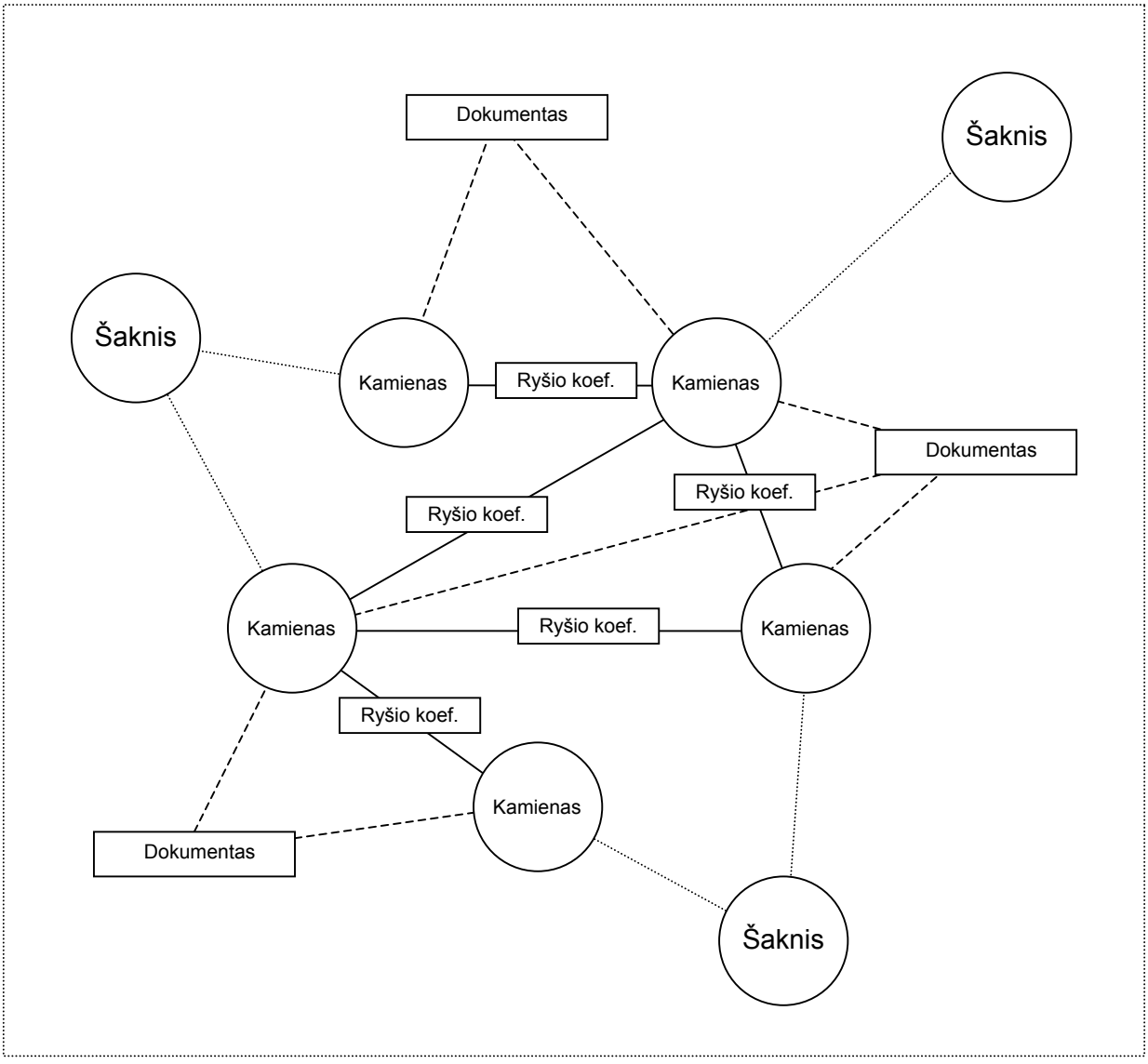
Dėl aukščiau išvardintų abiejų grafų trūkumų nei vienas automatiniam klasifikavimui nėra tinkamas. Tačiau apjungus abu variantus į viena struktūrą gauname grafą kurį galima panaudoti praktiškai. Jo schema pavaizduota 5.3.1 pav. (pavaizduotas grafas, kuriame integruoti kelių dokumentų semantiniai vaizdai).

Šiame grafe sąvokas atitinka kamienai, tačiau jo viršūnėse galima saugoti arba žodžio šaknį arba kamieną. Viršūnės, kuriose saugoma šaknis, yra sujungtos su viršūnėmis, kuriose saugomi kamienai su tokia šaknimi. Viršūnės su kamienais yra sujungtos tarpusavyje lankais su svoriais, kurie nusako sąvokų tarpusavio ryšio stiprumą. Su jomis susiejame ir dokumentus, kurių semantiniuose vaizduose sutinkamos tokios sąvokos.

Taip apibrėžto grafo privalumai yra šie:

- Kadangi grafe yra saugomos žodžių šaknys analizės metu įmanoma grupuoti žodžius pagal jų šaknis ir taip parinkti optimalų sąvokų skaičių
- Paieškos metu žodžių šaknų saugojimas sumažina reikalingų palyginimų skaičių, kadangi šaknų visada bus mažiau negu kamienų
- Skirtingos sąvokos su vienoda šaknimi („įstatymas“, „pastatas“) yra atskiriamos
- Grafas gaunasi ne „plokščias“, o dviejų lygių (šaknys - kamienai), kas suteikia galimybę papildomoms analizės galimybėms

Trūkumu galima įvardinti didesnį informacijos kiekį kurį reikės išsaugoti, tačiau privalumai tai kompensuoja.



5.1.1 pav. Kombinuoto grafo, kurio viršūnėse yra žodžio šaknys arba kamienai, schema.

6 Tekstų analizės algoritmai ir principai

Automatinio tekstų klasifikavimo uždavinys reikalauja, kad nebūtų vykdomas išankstinis „apmokymas“. Todėl pagrindine priemone tekstų analizei yra pasirenkami statistiniai metodai. Žemiau aprašomi algoritmai ir principai, kurie sprendžia automatinio dokumentų klasifikavimo uždavinį remiantis šiais metodais.

6.1 Tekstų analizės principai

Dokumentų klasifikavimo yra pakankamai subjektyvus uždavinys, kadangi net ir automatinio klasifikavimo rezultatus revizuoja atitinkamų sričių ekspertai, kurie ir nustato ar dokumentas klasifikuotas teisingai ar ne. Tokiais atvejais kai ekspertų nuomonės išsiskiria, nebeįmanoma net ir subjektyviai nustatyti kas yra teisinga. Darbo tikslas sukurti algoritmą maksimaliai atsiribojantį nuo subjektyvių nuomonių. Todėl pagrindinį principą, kuriuo remiantis bus vykdomas tolimesnis darbas suformuluosime taip: **nustatyti, kokiai kategorijai (kategorijoms) turi būti priskirtas duotasis tekstas, galima pagal šio teksto turinį, t.y. žodžius sutinkamus šiame tekste**.

Šio teiginio teisingumas neabejotinai gali būti užginčytas, tam nesunku sugalvoti pavyzdžių: pvz. enciklopedijos straipsnyje apie sieros rūgštį gali nei karto nebūti paminėtas žodis „chemija“, kaip tokiu atveju priskirti straipsnį šiai kategorijai? Taipogi būtų be galo sunku pagal teksto turinį nustatyti jo žanrą, pvz. pagal šio darbo turinį nustatyti, jog tai „mokslinis-tiriamasis darbas“. Tam reikėtų pasitelkti eksperto „meta“ žinias, kurios išeina už vieno atskirai paimto teksto ribų. Tačiau šios „meta“ žinios buvo įgytos analizuojant kitus tekstus, perskaičiuos chemijos vadovėlį, kuriame dažnai kartojasi žodžiai „chemija“ ir „rūgštis“. Tada grandinėlė „chemija“ – „rūgštis“ – „sieros rūgštis“ tampa viso labo išsiplėtusios duomenų bazės padariniu. Sudėtingesnė situacija su žanru, tačiau jei turėsime tekstą kuris aiškina kas tai yra „mokslinis-tiriamasis darbas“, kokie žodžiai būdingi tokiems darbams, tai greičiausiai dažnai šiame tekste sutinkami dažnai sutinkami žodžiai „klasifikavimas“, „metodas“ pateks į tą sąrašą. Taigi galima laikyti, jog auščiau suformuluotas pradinis principas yra teisingas, o jį papildyti galima tokiu: **kuo daugiau tekstų yra analizuojama, tuo tikslesnį klasifikavimą galima atlikti**.

6.2 Tekstą apibūdinančių sąvokų nustatymas

Tekstams apibūdinti naudosime kombinuotąjį grafą (žr. 5.3), kuriame sąvokas atitinka žodžių kamienai. Remiantis aukščiau suformuluotu principu, žinome, kad tekste

galima rasti tokius žodžius, kurie jį apibūdintų. Taipogi nustatyta, jog sąvoka ≠ žodis. Taigi reikalingas algoritmas, kuris spręstų tokį uždavinį: iš duotojo teksto išskirti žodžius, kurie gali būti tekstą apibūdinančiomis sąvokomis. Šio algoritmo esmė yra kiek ir kokius žodžius jis išskirs.

Darome prielaidą, jog geriausiai dokumentą apibūdina sąvokos, kurios dažnai pasikartoja. Toks postulatas yra neišvengimas, kadangi jokios papildomos semantinės informacijos apie tekstą mes neturime. Tačiau jo teisingumas daugelyje atvejų yra gana akivaizdus ir naudojamas kituose klasifikavimo metoduose (žr. 2 skyrių).

Kadangi tekstuose be žodžių galime sutikti kitokius darinius (skaičius, sutrumpinimus, datas) reikalinga apibrėžti kuriuos iš jų analizuosime. Žodžiu laikysime iš lietuviškos iš abėcėlės sudarytas simbolių sekas, netrumpesnes nei 3 simboliai. Taipogi neanalizuosime žodžių, kurie gali dažnai pasitaikyti, tačiau neturi semantinės prasmės (žr. 4.1 skyrių). Tokių žodžių sąrašą reikia sudaryti iki teksto analizės.

Remiantis aukščiau išdėstytais samprotavimais suformuluojame algoritmą, kuris paprastumo dėlei yra suskaidomas į tris dalis. Algoritmo pradinis duomuo yra tekstas, darbo rezultatas yra dokumentą apibūdinančių sąvokų sąrašas ir jų šaknų sąrašas.

I algoritmo dalis

- Sudarome visų analizuojamų teksto žodžių sąrašą
- Su kiekvienu analizuojamu žodžiu darome:
    - Randame žodžio šaknį ir kamieną
    - Tikriname, ar tokia šaknis jau įtrauktą į sąrašą
        - Jei tokia šaknies nebuvo įtraukta:
            - įtraukiame šaknį į sąrašą
            - susiejame su ja žodžio kamieną
            - kamienui suteikiame savybes:
                - skaitliukas, nusakantis tokių kamienų pasikartojimą
                - skaitliukas, nusakantis kamieno poziciją tekste
                - kandidatas į vienaskaitos vardininkus, priskiriamas analizuojamas žodis
        - Jei tokia šaknis jau buvo įtraukta:
            - tikriname, ar toks kamienas jau susietas su šaknimi:
                - Jei susietas
                    - Pasikartojimų skaitliuką padidiname 1

        - Pozicijos skaitliukui pridedame kamieno poziciją
        - Pagal vienaskaitos vardininko algoritmą nustatome, ar išsaugotas ar analizuojamas žodis yra priskirimas
    - Jei nesusietas
        - susiejame kamieną su šaknimi
        - kamienui suteikiame savybes
            - skaitliukas, nusakantis tokių kamienų pasikartojimą
            - skaitliukas, nusakantis kamieno poziciją tekste
            - kandidatas į vienaskaitos vardininkus, priskiriamas analizuojamas žodis
- baigiame ciklą

Algoritmo I dalies tikslas – sudaryti sutinkamų žodžių šaknų sąrašą ir su šaknimi susieti žodžių kamienus. Kamienams priskiriamos savybės nusako jų pasikartojimų kiekį, pozicijų tekste sumą, bei labiausiai tikėtiną žodžio vienaskaitos vardininko formą.

II algoritmo dalis

- Nustatome slenkstį $sl_{gr} = 0{,}3$
- Su kiekviena šaknimi iš sąrašo darome:
    - nustatome visų su šaknimi susietų kamienų pasikartojimų kiekį *vk*
    - pašaliname visus susietus su šaknimi kamienus, kuriems sąlyga $\frac{pk}{vk} < sl_{gr}$ yra tenkinama, kur *pk* yra kamieno pasikartojimų kiekis.
    - jei turi būti buvo pašalinti visi kamienai, paliekamas vienas, kurio santykis $\frac{pk}{vk}$ didžiausias
    - likusiems kamienams tolygiai pridedami išmestųjų kamienų pasikartojimų kiekiai
- baigiame ciklą

Algoritmo II dalies tikslas yra pašalinti retai pasitaikančius žodžių kamienus, ir sustiprinti dažniau pasitaikančius. To reikia, nes tekstuose dažnai pasitaiko giminingi žodžiai (pvz. „mokykla“, „mokytojas“, „mokinys“) kurių pasikartojimas nėra vienodai dažnas, ir kiekvieną iš jų traktuoti kaip atskirą sąvoką nėra prasminga. Todėl paliekami tik tokie kamienai, kurių pasikartojimų santykis su visais tokios šaknies žodžiais yra didesnis nei 30%. Tai apriboja su šaknim susietų kamienų kiekį iki trijų. Taipogi svarbu neprarasti informacijos dėl giminingų žodžių pasikartojimo, kuri bus reikalinga tolimesnei analizei. Tam pašalintų kamienų pasikartojimų suma lygiomis dalimis išdalinama pasilikusiems kamienams.

III algoritmo dalis

- Nustatome slenkstį $sl_{sk} = \log_{10}(vz) + 3$, kur *vz* – visų analizuojamų teksto žodžių skaičius.
- Nustatome slenkstį $sl_{\min} = vz \cdot 0{,}15$, kur *vz* – visų analizuojamų teksto žodžių skaičius.
- Nustatome slenkstį $sl_{\max} = vz \cdot 0{,}3$, kur *vz* – visų analizuojamų teksto žodžių skaičius
- Surūšiuojame visus kamienus pagal pasikartojimų kiekį mažėjimo tvarka, taip kad $pk_i > pk_{i+1}$, kur pk_i - atitinkamo kamieno pasikartojimų kiekis
- Randame tokį minimalų *n*, kad būtų tenkinamos nelygybės $\begin{cases} n > sl_{sk}, \\ \sum_{i=1}^{n} pk_i > sl_{\min} \end{cases}$,

  arba nelygybė $\sum_{i=1}^{n} pk_i > sl_{\max}$
- Iš surūšiuoto kamienų sąrašo pašaliname visus kamienus, kurių koeficientas didesnis už nustatytą *n*
- Iš sąrašo pašalinamos šaknys, su kuriomis neliko susietų kamienų

Algoritmo III dalies tikslas pašalinti retai pasitaikančių žodžių kamienus ir šaknis. Rezultate gaunamas šaknų ir kamienų sąrašas ir yra aukščiau reikalaujamas tikslas.

Kiek bus atrinkta dokumentą apibūdinančių sąvokų apsprendžia slenksčiai sl_{sk}, $sl_{\max}$, $sl_{\min}$. Pirmasis slenkstis nusako kiek sąvokų turi apibūdinti dokumentą ir priklauso nuo viso dokumento žodžių kieko . Slenksčiai $sl_{\min}$ ir $sl_{\max}$ skirti koreguoti pradinį slenkstį priklausomai nuo dokumento turinio. Pavyzdžiui jei dokumente yra daug skirtingų,

tačiau vienodai retai pasikartojančių sąvokų, jos bus įtrauktos nepaisant to, kad nustatytas sąvokų kiekis viršytas. Iš kitos pusės, jei dokumente žymiai dominuoja viena ar kelios sąvokos, tai dokumentą apibūdinančių sąvokų kiekis bus mažesnis nei numatyta. Keičiant slenksčių skaitines vertes galima gauti skirtingus dokumentus apibūdinančių sąvokų kiekius. Duotieji parametrai buvo nustatyti kaip optimalūs vykdant eksperimentus (žr. 8 skyrių), tačiau gali varijuoti priklausomai nuo analizuojamų dokumentų struktūros ar turinio.

6.3 Ryšių tarp sąvokų apskaičiavimas

Kaip parodyta 4.1 skyriuje, sąvoką gali sudaryti keli žodžiai. Tačiau naudojantis tik statistiniais metodais neįmanoma nustatyti ar sąvoka susideda iš vieno ar iš kelių žodžių. Visgi galima apskaičiuoti dviejų atrinktų sąvokų tarpusavio ryšio koeficientą, kuris parodytų, kiek tos dvi sąvokos yra susijusios. Darome prielaidą, jog sąvokų kurios yra susijusios pozicijos tekste bus panašios, t.y. eis viena po kitos ar pasirodys viename sakinyje ir panašiai (pvz. naudojant sąvoka „transporto priemonė“ dvi atskiros sąvokos „transportas“ ir „priemonė“ yra rašomos šalia). Tada formulė ryšiui tarp sąvokų k_i ir k_j atrodo taip:

$$r_{koef} = \frac{vz - \left| \frac{ak_i}{pk_i} - \frac{ak_j}{pk_j} \right|}{vz}$$, kur

vz – dokumento žodžių kiekis,

ak_i, ak_j – sąvokų k_i ir k_j pozicijų tekste numerių suma,

pk_i, pk_j – sąvokų k_i ir k_j pasikartojimo kiekis, vk_i ir vk_j.

Matome, kad ryšio koeficientas gali įgyti reikšmes intervale (0, 1). Akivaizdu, kad jei sąvokų k_i ir k_j pasikartojimo dažniai yra panašūs ir pozicijų numerių sumos panašios (t.y. sąvokos rašomos viena šalia kitos), tai ryšio koeficientas artėja į 1, kas rodo glaudų sąvokų ryšį. Taipogi pastebime, kad sąvokos apibūdinančios vieną dokumentą visada turės tarpusavio ryšį su tam tikru koeficientu. Turėdami šaknų, sueitų su kamienais sąrašą, taipogi kamienus tarpusavyje sujungtus ryšiais, kombinuoto grafo duomenų struktūrą (dokumento semantinį vaizdą) .

6.4 Dokumentų semantinių vaizdų integravimas

Pavienius dokumentų semantinius vaizdus integruojant į vieną duomenų struktūrą įvykdome antrą užsibrėžtą klasifikavimo principą (žr. 6.1 skyrių). Bendrajai struktūrai atvaizduoti taipogi naudosime kombinuotą grafą. Tuomet kiekvieno integruojamo dokumento grafui vykdome tokį algoritmą:

- Jei dar neegzistuoja, įtraukiame į bendrąją struktūrą šaknis, kurios yra integruojamo dokumento grafe
- Jei dar neegzistuoja, įtraukiame į bendrąją struktūrą kamienus, kurie yra integruojamo dokumento grafe
- Pakoreguojame savybes jau egzistuojančių šaknų ir kamienų, jei jie yra integruojamo dokumento grafe (pridedame žodžių kiekius, pozicijų sumas)
- Ryšius tarp kamienų integruojamame grafe
    - Jei dar neegzistuoja, perkeliame į bendrąją struktūrą
    - Jei egzistuoja, nustatome naują ryšio koeficientą r_{koef} pagal formulę

$$r_{koef} = \frac{ds \cdot r_{sen} + r_{\text{int}}}{ds + 1}, \text{ kur}$$

ds – integruotų dokumentų, kuriuose toks ryšys buvo, skaičius;

r_{sen} – egzistavęs ryšio koeficientas;

r_{int} – integruojamo ryšio koeficientas.

- Pridedame dokumentą apibūdinantiems kamienams ryšį su dokumentu. Ryšio savybė – dokumento ir kamieno ryšio stiprumas, apskaičiuojamas kaip kamieno pasikartojimų kiekio ir viso dokumento žodžių kiekio santykis.

Integravimo rezultatas – kombinuotas grafas, kuriame atsispindi visų analizuotų dokumentų klasifikavimo rezultatai. Jį galima naudoti kaip duomenų bazę paieškos sistemoms, o atlikus papildomą analizę kurti kitokias duomenų struktūras (žr. 6.6 skyrių).

6.5 Tekstų semantinių vaizdų palyginimas

Aukščiau aprašyta metodai įgalina ne tik priskirti tekstus kategorijoms, tačiau ir vykdyti atskirų dokumentų palyginimą. Tarkime turime dokumentus d_i ir d_j, tada procentinį dokumentų tarpusavio ryšį $dproc_{ij}$ paskaičiuojame pagal tokį algoritmą:

- Nustatome visas bendras d_i ir d_j apibūdinančias sąvokas $K = (k_1, ..., k_n)$

- Paskaičiuojame $dsav_{ij}$ pagal formulę

$$dsav_{ij} = \frac{\sum_{a=1}^{n}(1-\left|k_a d_i - k_a d_j\right|)}{n}, \text{ kur}$$

$k_a d_i$ – sąvokos k_a ir dokumento d_i ryšio koeficientas;

$k_a d_j$ – sąvokos k_a ir dokumento d_j ryšio koeficientas;

- Nustatome ryšių koeficientų sumą tarp aibės K sąvokų grafe d_i, pagal formulę:

$$RK_i = \frac{\sum_{a=1}^{n}\sum_{b=1}^{n} rk_{ab}}{n \cdot (n-1)/2}, \text{ kur}$$

rk_{ab} – ryšio koeficientas tarp sąvokų k_a ir k_b dokumente d_i.

- Nustatome ryšių koeficientų sumą tarp aibės K sąvokų grafe d_i, pagal formulę:

$$RK_j = \frac{\sum_{a=1}^{n}\sum_{b=1}^{n} rk_{ab}}{n \cdot (n-1)/2}, \text{ kur}$$

rk_{ab} – ryšio koeficientas tarp sąvokų k_a ir k_b dokumente d_j.

- Paskaičiuojame dokumentų tarpusavio procentinę priklausomybę $dproc_{ij}$ pagal formulę:

$$dproc_{ij} = dsav_{ij} \cdot 90 + \left|RK_i - RK_j\right| \cdot 10.$$

Taip apibrėžto dydžio $dproc_{ij}$ pagrindinę dalį sudaro sutampančios kategorijos (iki 90%), o ryšių tarp sąvokų koeficientų sutapimas tik nedidelę dalį (iki 10%). Tokios proporcijos buvo parinktos kaip pakankamai optimalios eksperimentų metu (žr. 8.1, 8.2 skyrius). Taipogi nesunku įsitikinti, jog $dproc_{ij} \neq dproc_{ji}$, kas yra teisinga savybė. Pvz. jei didelį dokumentą d_i apibūdina daug įvairių sąvokų, tai jis siejasi silpnai su mažu dokumentu d_j kuriame yra tik dalis pirmojo sąvokų. Tuo tarpu d_j siejasi su dokumentu d_i labai gerai, jei jo visos sąvokos patenka tarp daugelio d_i apibūdinančių sąvokų.

6.6 Algoritmas sąvokų medžio sudarymui

Grafo duomenų struktūra semantinių vaizdų sudarymui buvo pasirinkta dėl savo paprastumo ir galimybės pakankamai lanksčiai aprašyti dokumentus. Tačiau tokia struktūra nėra labai patogi vizualiam atvaizdavimui. Duomenų klasifikavimo sistemose katalogai vartotojams paprastai pateikiami kategorijų (sąvokų) medžio struktūroje (žr. 2.5 skyrių). Todėl iškyla natūralus klausimas kaip galima būtų kombinuotąjį grafą atvaizduoti medžiu.

Šio uždavinio sprendimas nėra trivialus, kadangi analizėje naudojami statistiniai metodai negali iki galo atskleisti sąvokų semantikos. Todėl siūlomas grafo pavertimo medžiu algoritmas yra tik vienas iš galimų sprendimo būdų, kurie negarantuoja idealaus rezultato bet kokiems pradiniams duomenimis. Tačiau tokio medžio generavimas gali būti prasmingas kaip pagrindas, reikalaujantis vėlesnių ekspertų korekcijų.

Prieš bandant nustatyti algoritmą spręsiantį nurodytą uždavinį, būtina suformuluoti kelis principus, kuriais vadovaujantis bus sudarinėjamas sąvokų medis:

- Kiekviena medžio viršūnė gali turėti *n* vaikų
- Dokumentai yra susiejami su medžio viršūnėmis
- Visi dokumentai, kurie buvo analizuojami, turi būti susieti bent su viena viršūne (t.y. turi būti galimybė medyje rasti visus dokumentus)
- Į medį turi patekti ne visos iš dokumentų išskirtos sąvokos, o tik dažniausiai pasitaikančios
- Laikysime, kad dažniau pasitaikančios sąvokos yra bendresnės už rečiau pasitaikančias

Paskutinysis principas yra labiausiai ginčytinas, kadangi nesunkiai galima sugalvoti kitokių pavyzdžių (pvz. įstatymuose žodis „straipsnis“ dažnai gali būti naudojamas žymiai dažniau nei pats žodis „įstatymas“, tačiau būtent ši sąvoka turėtų būti bendresnė). Visgi neturėdami jokios papildomos informacijos laikysime, jog šis teiginys yra teisingas.

Remiantis aukščiau išvardintais samprotavimais algoritmą formuluosime taip:

- Kiekvienai sąvokai s apskaičiuojame jos pasikartojimų dažnumo koeficientą s_{koef}

$$s_{koef} = \frac{dk - ds + 1}{sk \cdot ds}, \text{ kur}$$

sk – sąvokos pasikartojimų kiekis visuose dokumentuose;

ds – dokumentų, kuriuos apibūdina ši sąvoka kiekis;

dk – visų dokumentų kiekis

Nesunku įsitikinti, jog s_{koef} gali įgyti reikšmes intervale (0,...,1], ir kuo šis koeficientas bus mažesnis, bendresne šią sąvoką laikysime

- Kiekvieną sąvoką s priskirsime aibėms $A_i, i \in (1,..16)$ pagal s_{koef} taip:

$$s \in A_i \text{ , jei } a_{i-1} < s_{koef} \le a_i \text{ , kur}$$

$a_i = \frac{2^i}{pk}$, kur pk – visų sąvokų pasikartojimų suma;

Atlikę ši veiksmą sugrupuojame sąvokas į aibes A_i pagal jų pasikartojimų kiekius, dažniau pasikartojančios sąvokos patenka į aibes su mažesniais indeksais (šis priskyrimas plačiau komentuojamas skyriaus pabaigoje)

- Nustatome tokią ne tuščią aibę A_j, kurios indeksas j yra pats mažiausias ir joje esančias sąvokas priskiriame medžio šakniai kaip jos vaikus
- Visiems dokumentams nustatome dažniausiai pasikartojančias juos apibūdinančias sąvokas ir priskiriame medžio šakniai kaip vaikus (jei tokios sąvokos dar nėra priskirtos). Šis žingsnis užtikrina, kad medyje bus rasti visi dokumentai
- Su visais medžio šaknies vaikais rekursiškai vykdome tokius veiksmus:
    a) Nustatome su viršūnės sąvoka susijusias sąvokas
    b) jei susijusi sąvoka priklauso aibei A_{i+1} ir kelyje nuo einamosios viršūnės iki medžio šaknies dar nebuvo paminėta, įtraukiame į einamosios viršūnės vaikų sąrašą
    c) su kiekvienu viršūnės vaiku rekursiškai atliekame a)

Taip aprašytas algoritmas sudarys medį kuriame bus paminėtos visos sąvokos esančios grafe. Tačiau tai nėra gerai dėl dviejų priežasčių: medis gausis gilus ir šakotas, o dėl semantinės informacijos stokos daugelis medžio ryšių gali būti beprasmiški. Todėl galima įvesti kelis parametrus ribojančius sąvokų pasikeitimą:

- Viršūnės vaikais priskirti tik tokias sąvokas, kurių ryšio koeficientas r_{koef} (žr. 6.4 skyrių) būtų didesnis už tam tikrą reikšmę, pvz. 0,5; 0,75; 0,9.
- Apriboti rekursijos gylį, pvz. $j+1$; $j+2$, $j+5$.

Nustatyti optimalias koeficientų reikšmes visiems atvejams atrodo neįmanoma, jie labai priklauso nuo pradinių duomenų. Tačiau jei laikysime, jog prima yra suintegruojami visi dokumentai, o tik tada generuojamas kategorijų medis, tuomet ir toks metodo trūkumas gali būti laikomas priimtinu.

Sąvokos s priskyrimą aibei A_i galima pailiustruoti sąvokas atvaizdavus grafiškai polinėse koordinatėse. Laikysime, kad s_{koef} atitinka vektoriaus ilgį, o s_{pos} (vidutinė sąvokos pozicija dokumente) atitinką kampą ir yra apskaičiuojamas:

$$s_p = \frac{sp}{sk}$$, kur

sk – sąvokos pasikartojimų kiekis visuose dokumentuose;

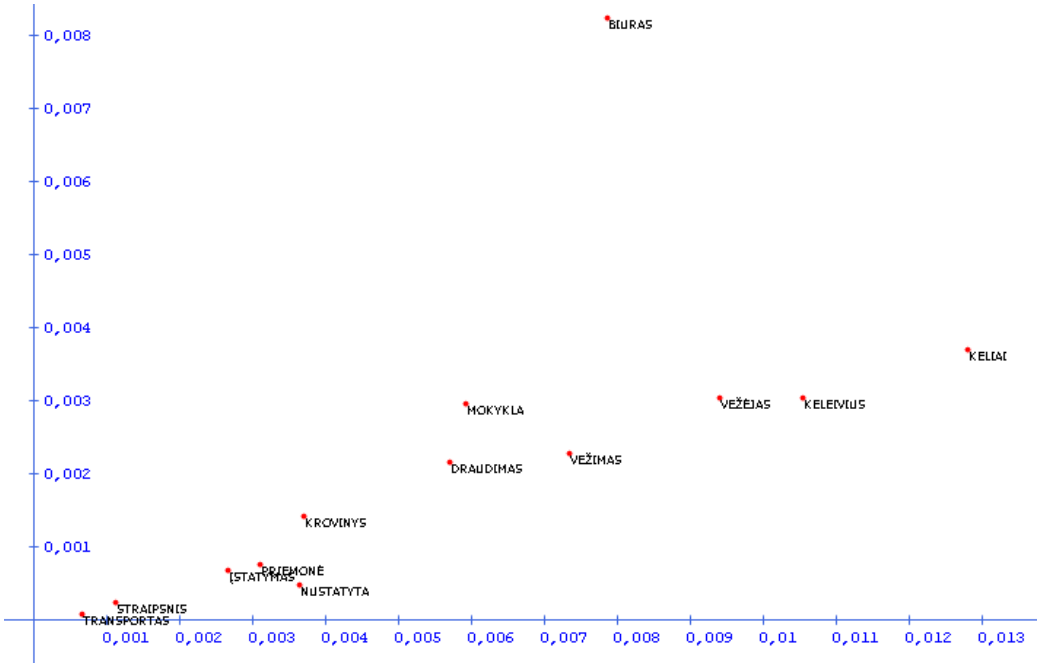
sp – sąvokos pozicijų suma visuose dokumentuose.

Tada polines koordinates sąvokai s skaičiuosime taip:

$$\begin{cases} x = s_{koef} \cdot \cos(\frac{s_p}{pk \cdot \pi}) \\ y = s_{koef} \cdot \sin(\frac{s_p}{pk \cdot \pi}) \end{cases}$$, kur

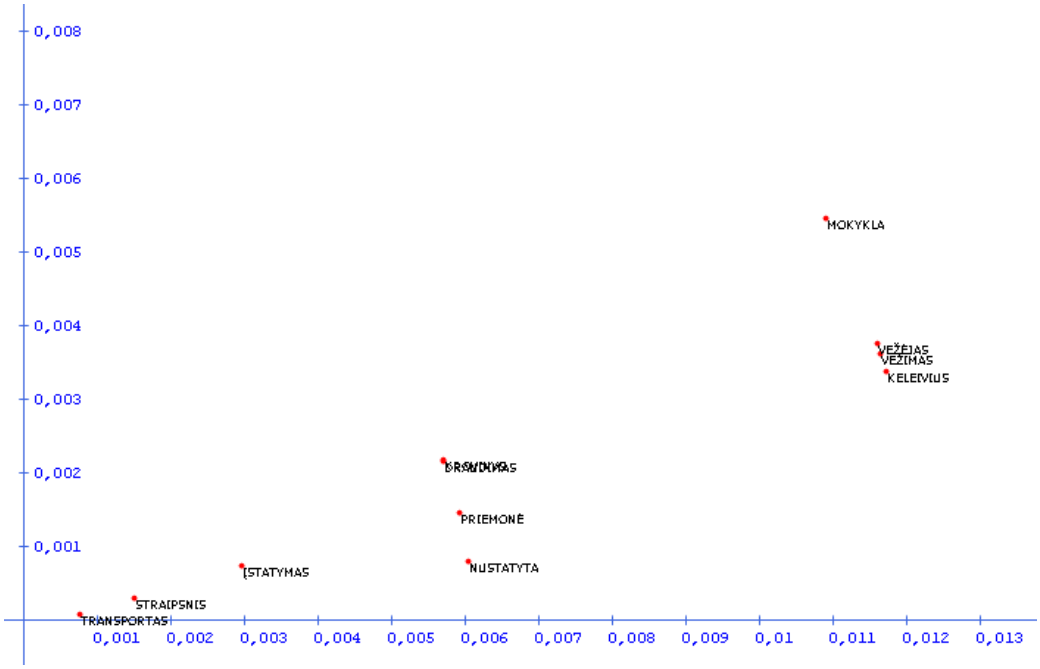
pk – visų sąvokų pasikartojimų suma.

Tokiu būdu sąvokos gali „išsibarstyti“ tik I ir II koordinačių plokštumos dalyje, artėjimas prie nulio nusako pasikartojimų dažnį (duotame kontekste sąvokos bendrumą), o kampas - sąvokų tarpusavio ryšį (kuo mažesnis kampo skirtumas tarp sąvokų, tuo jos artimesnės). Sąvokų atvaizdavimas polinėse koordinatėse pavaizduotas pav. 6.6.1.



6.6.1 pav. Sąvokos polinės koordinatėse

Kaip matome, sąvokos yra pasiskirsčiusios gana chaotiškai. Todėl jei įvesime tam tikrus slenksčius (žr. aukščiau $a_i = \frac{2^i}{pk}$) ir suapvalinsime vektorių ilgių reikšmes gausime taisyklingesnį vaizdą (pav. 6.6.2).



6.6.2 pav. Sąvokos polinės koordinatėse, sugrupuotos

Matome, kad sąvokos tokiu atveju išsidėsto ant koncentrinių apskritimų (centro koordinatės (0,0)). Kiekvieną apskritimą galima interpretuoti kaip atskirą sąvokų medžio lygį, apskritimuose su mažesniais spinduliais atsiduria bendresnės sąvokos, su didesniais – specifinės. Taipogi taip išdėsčios sąvokas galime nesunkiai pastebėti, kurios sąvokos tarpusavyje yra labiau susijusios (susiejimo lygį nusako kampas).

7 Algoritmų ir vartotojo sąsajų realizacijos

Aukščiau aprašyti algoritmai ir metodai buvo realizuoti kaip vieninga programų sistema, atitinkanti 3 skyriuje išdėstytą projektą. Žemiau pateikiami realizuotų modulių aprašai bei priimtų realizacinių sprendimų pagrindimas.

7.1 Realizacijos įrankiai

Visų programinių komponentų realizacijai pasirinkta Microsoft VB.NET platforma ir XML duomenų perdavimui bei saugojimui. Toks pasirinkimas nulemtas šių faktorių:

- šiuolaikiniai įrankiai leidžiantys realizuoti visą normą funkcionalumą
- programavimo paprastumas
- darbo kontekste nebūtina užtikrinti maksimalios sistemos veikimo spartos
- autoriaus patirtis dirbant VB.NET platforma

Pasirinkimo trūkumu galima įvardinti prisirišimą prie Microsoft operacinių sistemų, tačiau šio darbo rėmuose toks faktorius nėra lemiamas.

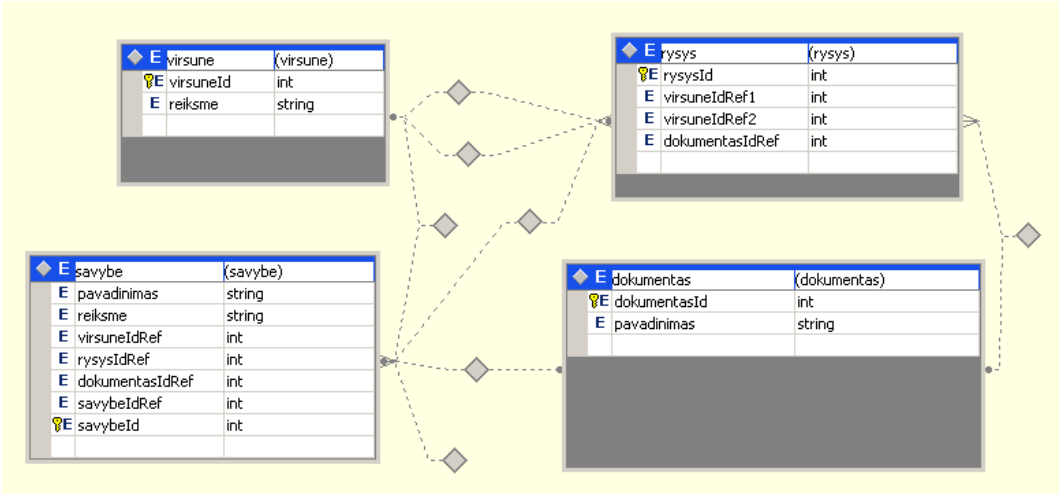
7.2 Semantinės duomenų struktūros specifikacija

Išanalizuotų tekstinių bylų duomenys ir pati klasifikavimo struktūra pagal projektą yra saugoma XML bylose. Norint užtikrinti galimybę įvairiems moduliams be trikdžių keistis informacija, būtina aprašyti XML schemą pagal kurią būtu formuojamos XML duomenų bylos. Vienu iš klasifikavimo uždavinio sprendimo realizacijos prioritetų yra įvardinta galimybė keisti semantinę struktūrą kurios pagrindu vykdomas klasifikavimas (darbo metu ieškant optimalaus klasifikavimo uždavinio sprendimo buvo išbandytos kelios duomenų struktūros, žr. 5 skyrių). Įmanomi variantai yra grafai, medžiai, tezaurai, semantinės gardelės. Tai apsprendžia pagrindinį reikalavimą XML schemai – lankstumą aprašant įvairias duomenų struktūras. Išanalizavus tokį poreikį išskirti keturi pagrindiniai XML schemos elementai:

- Viršūnė – sudaro identifikatorius ir reikšmė
- Ryšys – sudaro identifikatorius ir išoriniai raktai į viršūnes bei dokumentus
- Dokumentas – sudaro identifikatorius ir pavadinimas
- Savybė – sudaro identifikatorius, savybės pavadinimas, savybės reikšmė ir išoriniai raktai į viršūnes ryšius, dokumentus ir savybes

Viršūnės tarpusavyje ir su dokumentais susiejus ryšiais gauname bet kokią norimą duomenų struktūrą. Taipogi leidžiame kiekvienai viršūnei , ryšiui ar dokumentui priskirti norimą kiekį savybių, kurios savo ruožtu gali susidaryti iš norimo kiekio kitokių savybių.

Tai leidžia duomenų analizės metu neapsiriboti iš anksto numatytais parametrais, o juos pridėti ar atimti poreikiui esant. XML schemos diagrama pateikiama 7.2.1 pav.



7.2.1 pav. XML schemos diagrama

Toks duomenų aprašymo būdas suteikia reikalingą lankstumą trimo metu, tačiau reikalauja daugiau diskinės vietos duomenų saugojimui bei procesorinio laiko duomenų apdorojimui. Tačiau nepaisant trūkumų šis sprendimas atitinka iškeltus darbo prioritetus.

7.3 Analizės modulis

Dokumentų analizės modulio paskirtis – sudarinėti dokumentų, parašytų lietuvių kalba, semantinius vaizdus. Modulis realizuoja algoritmus aprašytus 6.1 ir 6.2 skyriuose, dokumentų semantinis vaizdas pateikiamas kaip kombinuotas šaknų-sąvokų grafas (žr. 5.3 skyrių).

Modulis realizuotas kaip komandinės eilutės aplikacija, be vartotojo sąsajos. Bylos kataloge būtinai turi būti išsaugota pradinių leksinių duomenų byla “duomenys.xml”. Pradiniai duomenys analizei pateikiami kaip paprastos tekstinės bylos, išsaugotos Unicode formate. Modulio darbo rezultatas yra XML bylos su dokumentų semantiniais vaizdais. Komandine eilute moduliui perduodami šie parametrai:

- kelias iki katalogo, kuriame išsaugotos bylos analizei (būtinas)
- kelias iki katalogo, kuriame reikia išsaugoti gautas XML bylas (būtinas)
- kelias iki katalogo, kuriame bus pastoviai laikomi dokumentai (būtinas)
- $sl_{\min}$ koeficientas, galimos reikšmės nuo 0 iki 1, jei nenurodomas pagal nutylėjimą lygus 0,15. Jei praleidžiamas, nurodoma reikšmė -1.

- $sl_{\max}$ koeficientas, galimos reikšmės nuo 0 iki 1, jei nenurodomas pagal nutylėjimą lygus 0,3.

Parametrai atskiriami tarpais, programos iškvietimo pvz.:

„analyser1.exe C:\Input\ C:\Output\ C:\Documents\ -1 0,25“

Paleidus programą, į konsolę yra išvedamas einamosios tekstinės bylos pavadinimas, vėliau analizuojamų žodžių kiekis. Įvykdžius analizę duomenys išsaugomi XML byloje ir jos pavadinimas yra išvedamas į konsolę.

Realizuojant šaknų – sąvokų grafą, aukščiau aprašyta XML struktūra buvo panaudota taip:

- Elemento „Viršūnė“ reikšmė – žodžio šaknis. Savybės:
  - „kiekZodziu“ – nusako kiek žodžių su tokia šaknimi buvo rasta dokumente
  - „susijesZodis“ – nusako sąvoką su tokia šaknimi. Savybės:
    - „kiekZodziu“ – nusako kiek sąvokų buvo rasta dokumente
    - „koksAtstumas“ – nusako pozicijų, kuriose buvo rastos sąvokos, skaitnę sumą
    - „zodzioBendratis“ – nusako galimą sąvokos vienaskaitos vardininką
- Elemento „Dokumentas“ reikšmė –dokumento pavadinimas. Savybės:
  - „zodziuDokumente“ – dokumento žodžių kiekis
  - „dokumentoPradzia“ – pirmi 300 dokumento simbolių (vėlesniam vizualizavimui supaprastinti)
- Elementas „Ryšys“ panaudotas ryšiams nustatyti tarp dviejų savybių „susijesZodis“, tarp savybės „susijesZodis“ ir dokumento. Savybės
  - „rysioKoeficientas“ – ryšio koeficientas tarp dviejų sąvokų (tarp dviejų savybių „susijesZodis“)
  - „zodisDokumente“ – sąvokos pasitaikymo dažnio santykis su bendru dokumento žodžių kiekiu (tarp dokumento ir savybės „susijęsZodis“)

7.4 Integravimo modulis

Dokumentų integravimo modulio paskirtis – apjunginėti į vieną struktūrą naujai išanalizuotų dokumentų semantinius vaizdus. Modulis realizuoja algoritmą aprašytą 6.3 skyriuje, dokumentų semantinis vaizdas pateikiamas kaip kombinuotas grafas (žr. 5.3 skyrių).

Modulis realizuotas kaip komandinės eilutės aplikacija, be vartotojo sąsajos. Duomenys integravimui pateikiami kaip XML dokumentai, paruošti analizės modulio. Modulio darbo rezultatas yra XML byla su integruota dokumentų semantinių vaizdų struktūra. Komandine eilute moduliui perduodami šie parametrai:

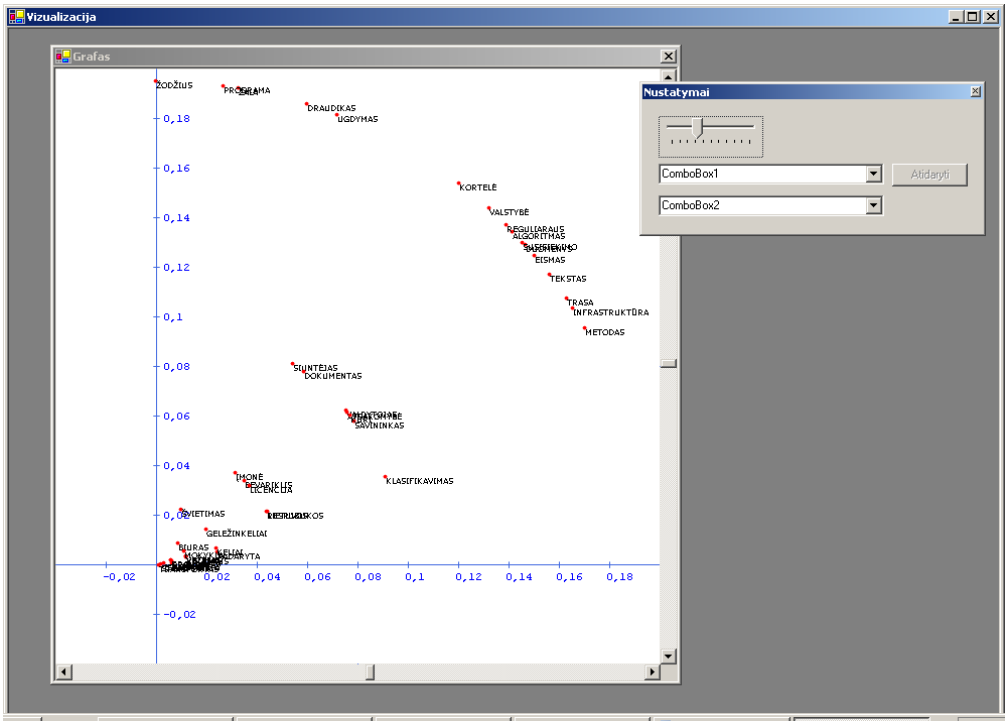
- kelias iki katalogo, kuriame išsaugotos bylos integravimui (būtinas)
- kelias iki XML bylos, kurioje yra išsaugota semantinių vaizdų struktūra (būtinas)

Parametrai nurodomi kabutėse, tarpusavyje atskiriami tarpais (programos iškvietimo pvz.: “integrator1.exe C:\Output\ C:\Structure\structure.xml”). Paleidus programą, į konsolę yra išvedamas einamosios integruojamos XML bylos pavadinimas. Prieš integruojant pirmą kartą būtina sukurti tuščią bendros semantinės struktūros XML bylą, kurios turinys:

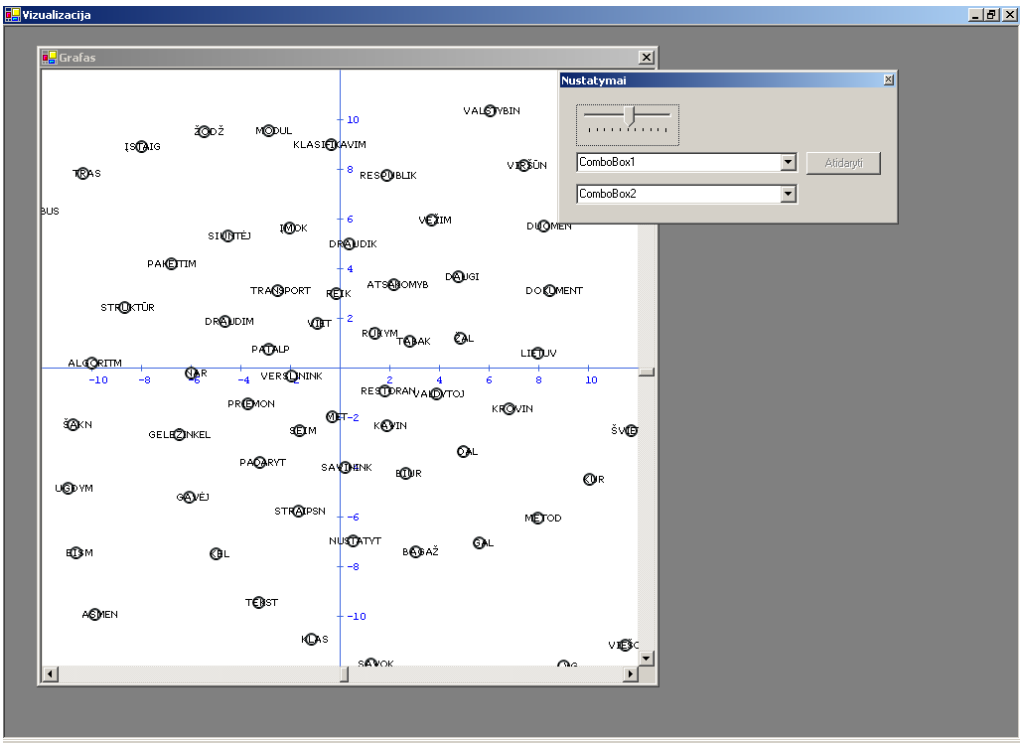
<?xml version="1.0" standalone="yes"?><struktura></struktura>

7.5 Grafinio duomenų atvaizdavimo modulis

Grafinio duomenų atvaizdavimo modulis skirtas grafų ir sąvokų atvaizdavimui tyrimų metu ir galutiniam vartotojui neprieinamas. Pav.7.5.1 ir 7.5.2 pavaizduoja grafinio modulio langus.



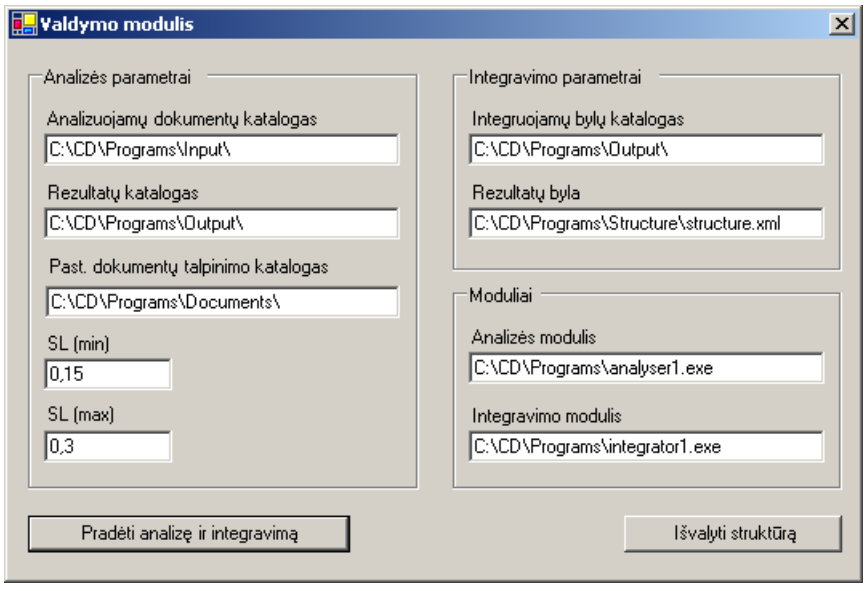
7.5.1 pav. Duomenų vizualizavimo modulis



7.5.2 pav. Duomenų vizualizavimo modulis

7.6 Valdymo modulis

Valdymo modulis („ControlUnit.exe“) yra skirtas duomenų analizės bei integravimo proceso valdymui. Jame nurodoma kokius modulius vykdyti, kokius parametrus jiems perduoti. Analizės ir integravimo procesas pardedamas paspaudus mygtuką „Pradėti analizę ir integravimą“. Mygtukas „Išvalyti struktūrą“ skirtas tuščios bendros semantinės struktūros XML bylos sukūrimui. Modulio langas pavaizduotas 7.6.1 pav.



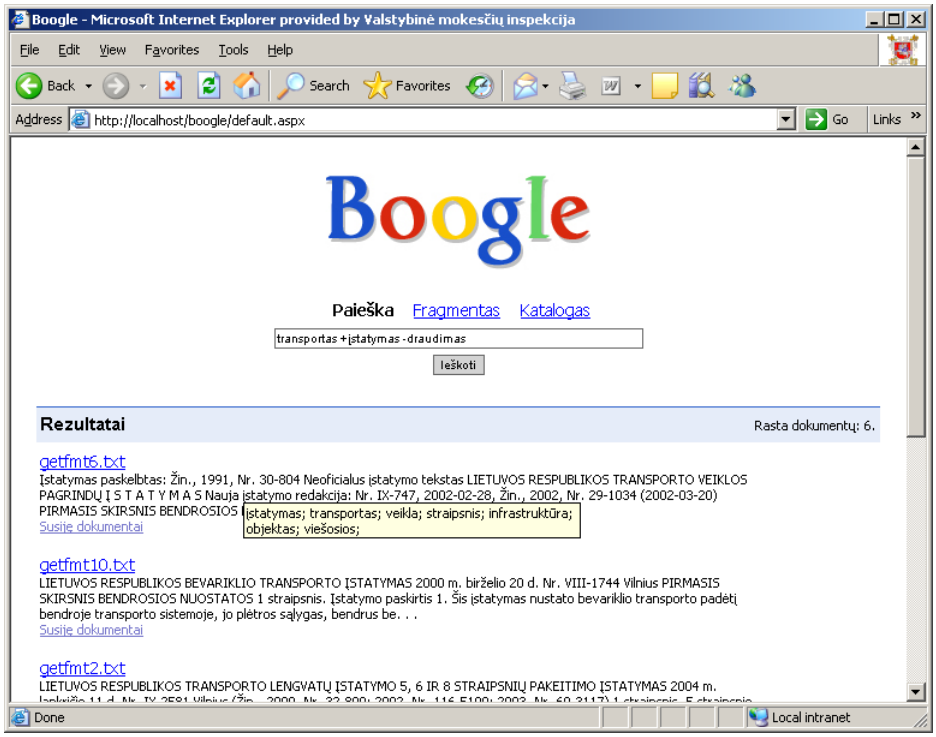
7.6.1 pav. Valdymo modulis

7.7 Web sąsaja

Praktiniam aprašytos klasifikavimo sistemos panaudojimui iliustruoti sukurta Web sąsaja, leidžianti vartotojams ieškoti dokumentų pagal raktinius žodžius, tekstų fragmentus arba naršyti po sukurtą kategorijų medį. Žemiau pateikiami šios sistemos aprašai.

7.7.1 Paieška pagal raktinius žodžius

Paprasčiausia dokumentų paieškos forma Web sąsajoje – paieška pagal raktinius žodžius. Jos langas atsidaro pirmas (pav. 7.7.1.1).



7.7.1.1 pav. paieška pagal raktinius žodžius

Operatorius “+“ nurodytas prieš raktinį žodį reiškia, jog tokia sąvoka būtinai turi būti tarp apibūdinančių dokumentą sąvokų. Operatorius “-“ nurodytas prieš raktinį žodį reiškia, jog tokia sąvoka negali būti tarp apibūdinančių dokumentą sąvokų. Jei operatorius nenurodytas, tai sąvoka gali būti tarp apibūdinančių dokumentą sąvokų. Paieškos algoritmas veikia tokiu principu:

- Randami susiję dokumentai su sąvokomis kurios nurodytos be operatoriaus ir su operatorium „+“
- Iš rezultatų pašalinami tie dokumentai, kurių neapibūdina sąvokos pažymėtos operatoriumi „+“

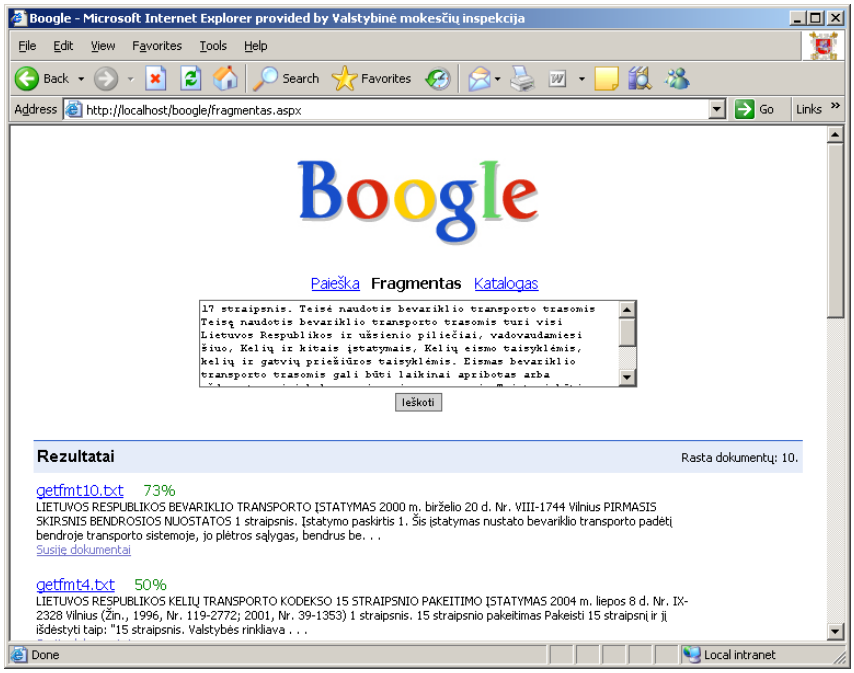
- Iš rezultatų pašalinami tie dokumentai, kuriuos neapibūdina sąvokos pažymėtos operatoriumi „-“
- Rezultatų sąrašas surūšiuojamas pagal sąvokų ir dokumentų ryšių koeficientus mažėjimo tvarka (labiausiai su paieškos žodžiais susiję dokumentai atsiduria viršuje)

Paieškos rezultatas pavaizduojamas sąrašu, kurio įrašo pradžioje nurodomas rasto dokumento pavadinimas (paspaudus dokumentas atsidaro naujame lange). Toliau rodoma dokumento teksto pradžia (palaikius pelės žymeklį virš jos rodomos su dokumentu susietos sąvokos), žemiau rodoma nuoroda „Susiję dokumentai“. Paspaudus šią nuorodą vykdoma paieška dokumentų, kurie galėtų būti susiję su duotuoju dokumentu (pav. 7.7.1.2). Paieška vykdoma pagal skyriuje 6.5 aprašytą algoritmą, žaliai pavaizduotas $dproc_{ij}$ dydis.

7.7.1.2 pav. dokumentų, susijusių su rastu dokumentu, paieška

7.7.2 Paieška pagal teksto fragmentą

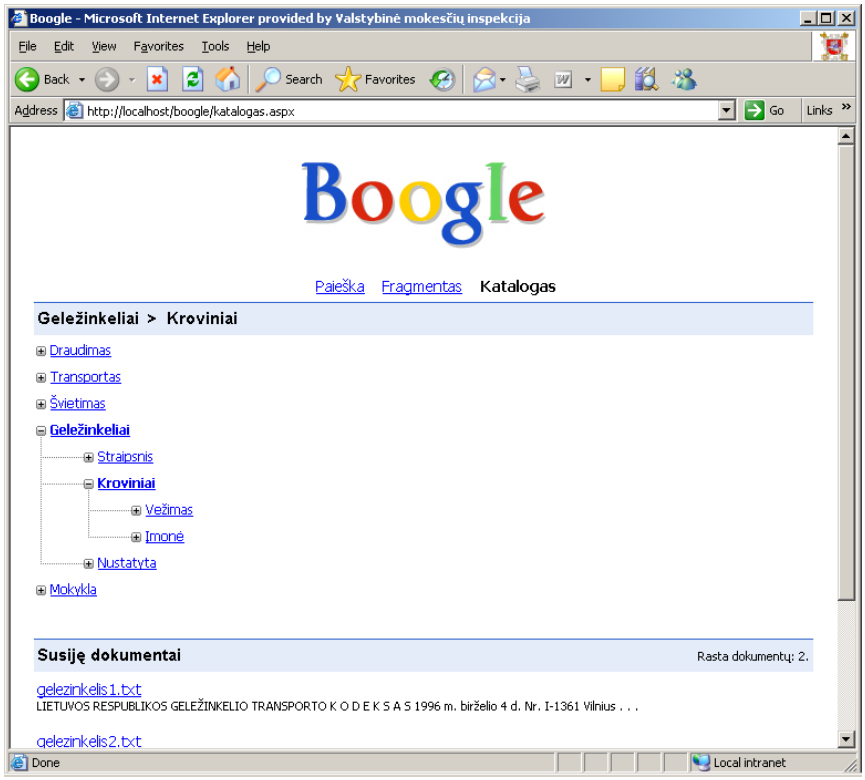
Šiame lange (pav. 7.7.2.1) leidžiama įvesti teksto fragmentą. Paieška vykdoma pagal 6.5 skyriuje aprašytą algoritmą, paieškos rezultatų atvaizdavimas analogiškas aprašytam 7.7.1 skyriuje.



7.7.2.1 pav. dokumentų, susijusių su nurodytu fragmentu, paieška

7.7.3 Sąvokų medis

Šiame lange (pav. 7.7.3.1) atvaizduojamas pagal algoritmą aprašytą 6.6 skyriuje sukurtas sąvokų medis. Paspaudus ant jo viršūnės apačioje rodomi su viršūne susiję dokumentai.



7.7.3.1 pav. sąvokų medis

8 Eksperimentai

Pagrindinius duomenis atliekant eksperimentus sudarė transporto įstatymai, tačiau siekiant teisingau įvertinti algoritmų darbo rezultatus, buvo panaudoti ir dokumentai, kurie tik iš dalies siejosi su transporto ir teisės tematika, arba visai nesisiejo. Viso įvairiais tyrimo etapais analizei buvo panaudota daugiau nei 50 skirtingų dokumentų. Žemiau pateikiami eksperimentų atliktų su aprašytais algoritmais rezultatai. Dokumentai ir juos apibūdinančios sąvokos.

8.1 Dokumentus apibūdinančios sąvokos

Lentelėje pateikiami kai kurių analizuotų dokumentų apibūdinančios sąvokos.

Nr.	Dokumento pavadinimas	Apibūdinančios sąvokos	Dydis (KB)
1.	LIETUVOS RESPUBLIKOS BEVARIKLIO TRANSPORTO ĮSTATYMAS 2000 m. birželio 20 d.	bevariklis; transportas; įstatymas; priemonė; trasa; keli; eismas;	38
2.	Neoficialus įstatymo tekstas LIETUVOS RESPUBLIKOS BEVARIKLIO TRANSPORTO ĮSTATYMAS 2000 m. birželio 20 d.	įstatymas; bevariklis; transportas; priemonė; trasa; keli; eismas;	42
3.	LIETUVOS RESPUBLIKOS TRANSPORTO VEIKLOS PAGRINDŲ ĮSTATYMAS	Lietuvos; respublikos; transportas; įstatymas; straipsnis; priemonė; nustatyta;	12
4.	1991, Nr. 30-804 Neoficialus įstatymo tekstas LIETUVOS RESPUBLIKOS TRANSPORTO VEIKLOS PAGRINDŲ Į S T A T Y M A S	įstatymas; transportas; veikla; straipsnis; infrastruktūra; objektas; viešosios;	39
5.	LIETUVOS RESPUBLIKOS TRANSPORTO VEIKLOS PAGRINDŲ Į S T A T Y M A S Nauja įstatymo redakcija	įstatymas; Lietuvos; transportas; straipsnis; vežėjas; infrastruktūra; valstybė;	59
6.	LIETUVOS RESPUBLIKOS TRANSPORTO PRIEMONIŲ SAVININKŲ IR VALDYTOJŲ CIVILINĖS ATSAKOMYBĖS PRIVALOMOJO DRAUDIMO ĮSTATYMAS 2001 m. birželio 14 d.	transportas; priemonė; savininkas; valdytojas; atsakomybė; draudimas; padaryta; biuras;	102
7.	LIETUVOS RESPUBLIKOS TRANSPORTO PRIEMONIŲ VALDYTOJŲ CIVILINĖS ATSAKOMYBĖS PRIVALOMOJO DRAUDIMO Į S T A T Y M A S 2001 m. birželio 14 d. Nauja redakcija	transportas; priemonė; draudimas; žala; padaryta; biuras; draudikas; narė;	179
8.	LIETUVOS RESPUBLIKOS KELIŲ TRANSPORTO KODEKSAS 1996 m. lapkričio 19 d.	keli; transportas; keleivius; krovinys; vežimas; priemonė; nustatyta; vežėjas;	68
9.	Neoficialus kodekso tekstas LIETUVOS RESPUBLIKOS KELIŲ TRANSPORTO KODEKSAS 1996 m. lapkričio 19 d.	keli; transportas; straipsnis; keleivius; krovinys; vežimas; nustatyta; vežėjas;	98
10.	Automatinis dokumentų klasifikavimas semantinių struktūrų pagrindu	dokumentas; klasifikavimas; tekstas; metodas; žodžius; algoritmas; šaknys; duomenys; sąvoka; keli;	132

8.2 Dokumentų palyginimai tarpusavyje

Kaip aprašyta 6.5 skyriuje, išanalizuotus dokumentus įmanoma palyginti tarpusavyje (semantinių dokumentų vaizdų palyginimas). Pateikiame eksperimentų rezultatus kai kuriems dokumentams. Lentelėje kiekvienam dokumentui nurodomi trys labiausiai susiję dokumentai ir jų susiejimo procentas.

Nr.	Dokumentas	Susiję dokumentai	Sus. proc.
1.	LIETUVOS RESPUBLIKOS BEVARIKLIO TRANSPORTO ĮSTATYMAS 2000 m. birželio 20 d.	Neoficialus įstatymo tekstas LIETUVOS RESPUBLIKOS BEVARIKLIO TRANSPORTO ĮSTATYMAS 2000 m. birželio 20 d.	99
		LIETUVOS RESPUBLIKOS TRANSPORTO VEIKLOS PAGRINDŲ ĮSTATYMAS	40
		LIETUVOS RESPUBLIKOS KELIŲ TRANSPORTO KODEKSAS	39
2.	LIETUVOS RESPUBLIKOS TRANSPORTO VEIKLOS PAGRINDŲ Į S T A T Y M A S Nauja įstatymo redakcija	LIETUVOS RESPUBLIKOS TRANSPORTO VEIKLOS PAGRINDŲ ĮSTATYMO 1 IR 2 STRAIPSNIŲ PAKEITIMO IR ĮSTATYMO PAPILDYMO TREČIUOJU SKIRSNIU BEI PRIEDU ĮSTATYMAS	54
		LIETUVOS RESPUBLIKOS TRANSPORTO VEIKLOS PAGRINDŲ ĮSTATYMAS	53
		1991, Nr. 30-804 Neoficialus įstatymo tekstas LIETUVOS RESPUBLIKOS TRANSPORTO VEIKLOS PAGRINDŲ Į S T A T Y M A S	53
3.	LIETUVOS RESPUBLIKOS TRANSPORTO PRIEMONIŲ VALDYTOJŲ CIVILINĖS ATSAKOMYBĖS PRIVALOMOJO DRAUDIMO Į S T A T Y M A S 2001 m. birželio 14 d. Nauja redakcija	LIETUVOS RESPUBLIKOS TRANSPORTO PRIEMONIŲ SAVININKŲ IR VALDYTOJŲ CIVILINĖS ATSAKOMYBĖS PRIVALOMOJO DRAUDIMO ĮSTATYMAS 2001 m. birželio 14 d. (pirmoji redakcija)	59
		LIETUVOS RESPUBLIKOS TRANSPORTO PRIEMONIŲ SAVININKŲ IR VALDYTOJŲ CIVILINĖS ATSAKOMYBĖS PRIVALOMOJO DRAUDIMO ĮSTATYMAS 2001 m. birželio 14 d. (antroji redakcija)	59
		LIETUVOS RESPUBLIKOS TRANSPORTO PRIEMONIŲ SAVININKŲ IR VALDYTOJŲ CIVILINĖS ATSAKOMYBĖS PRIVALOMOJO DRAUDIMO ĮSTATYMO 10, 14, 15, 18 STRAIPSNIŲ PAKEITIMO IR PAPILDYMO ĮSTATYMAS 2003 m. kovo 25 d.	34

4.	LIETUVOS RESPUBLIKOS GELEŽINKELIO TRANSPORTO K O D E K S A S 1996 m. birželio 4 d. (aktuali redakcija)	LIETUVOS RESPUBLIKOS GELEŽINKELIO TRANSPORTO K O D E K S A S 1996 m. birželio 4 d. (neaktuali redakcija)	86
		LIETUVOS RESPUBLIKOS KELIŲ TRANSPORTO KODEKSAS 1996 m. lapkričio 19 d.	47
		LIETUVOS RESPUBLIKOS KELIŲ TRANSPORTO KODEKSAS 1996 m. lapkričio 19 d.	47

8.3 Dokumentų paieška pagal ištraukas

Dokumentų semantinių vaizdų palyginimų įvertinimui galime pasinaudoti ir eksperimentu, kuomet pagal dokumento fragmentą ieškomas pats dokumentas. Eksperimentai atliekami taip: iš dokumentų sąrašo yra pasirenkamas vienas atsitiktinis dokumentas ir iš jo „iškerpamas“ atsitiktinis nurodyto ilgio (simboliais) fragmentas. Fragmentas yra padalinamas į tris atsitiktinio ilgio dalis, kurios yra sumaišomos tarpusavyje (šie veiksmai yra atliekami tam, kad eliminuoti sintaksinės paieškos galimybę, nes gautas fragmentas nebebus sutinkamas pradiniame dokumente). Gautam fragmentui sudaromas semantinis vaizdas, kuris yra lyginamas su išanalizuotų dokumentų semantiniais vaizdais. Rezultate gaunamas surūšiuotas pagal susiejimo procentą sąrašas dokumentų, kurie gali būti susiję su fragmentu. Įvertinsime ar tame sąraše yra dokumentas iš kurio buvo iškirptas pradinis fragmentas, ir jei yra, tai kurioje sąrašo vietoje jis atsidūrė.

Eksperimentai buvo atliekami su dviem dokumentų aibėm. Pirmoji sudaryta iš 17 transporto įstatymų, 2 švietimo įstatymų, 4 dokumentų nesusijusių su transportu ir įstatymais (viso 29 dokumentai, bendras dydis 1,7 MB). Fragmentams gauti buvo naudojama 23 dokumentai, kurių dydis ne mažesnis nei 8 KB (nepateko 4 įstatymai ir 2 nesusiję dokumentai). Reikia pastebėti, kad pirmoji aibė yra sudaryta taip, jog yra nepalanki tokiems eksperimentams atlikti, kadangi joje daug dokumentų kurie pagal prasmę yra labai panašūs vieni į kitus (to paties įstatymo skirtingo laiko aktualios redakcijos).

Lentelėje pateikiami eksperimentų rezultatai, kai kiekvieno ilgio fragmentams buvo atliekama po 1000 bandymų:

Fragmento ilgis	Sąraše pirmas (%)	Sąraše antras (%)	Rastas kitur (%)	Nerastas (%)	Vidutinė pozicija (kai rastas)	Vid. gauto sąr. ilgis
125	13,5	11,1	14,9	60,5	2,96	2,93
250	24,4	18,4	25,4	31,8	3,08	5,94

500	35,9	22,9	28,2	13	2,97	8,27
1000	38,3	24,7	31,2	5,8	2,93	10,28
2000	48,4	25,2	24,6	1,8	2,53	11,78
4000	50,8	26,6	22,5	0,1	2,33	13,96

Lentelėje pateikiami eksperimentų rezultatai, kai kiekvieno ilgio fragmentams buvo atliekama po 10000 bandymų:

Fragmento ilgis	Sąraše pirmas (%)	Sąraše antras (%)	Rastas kitur (%)	Nerastas (%)	Vidutinė pozicija (kai rastas)	Vid. gauto sąr. ilgis
125	13,97	10,53	15,08	60,42	3,03	3,21
250	24,14	16,46	24,74	34,66	3,05	5,68
500	33,62	22,57	29,55	14,26	2,97	8,29
1000	39,38	24,53	29,88	6,21	2,88	9,9
2000	46,1	25,19	26,92	1,79	2,69	11,65
4000	51,81	25,94	22,08	0,17	2,29	13,68

Pirmoji dokumentų aibė buvo papildyta 8 dokumentais nesusijusiais su transportu ir įstatymais (tekstai rasti MIF svetainėje) ir taip gauta antroji aibė. Fragmentams gauti buvo naudojami 32 dokumentai, kurių dydis ne mažesnis nei 8 KB (nepateko 4 įstatymai ir 3 nesusiję dokumentai).

Lentelėje pateikiami eksperimentų rezultatai, kai kiekvieno ilgio fragmentams buvo atliekama po 1000 bandymų:

Fragmento ilgis	Sąraše pirmas (%)	Sąraše antras (%)	Rastas kitur (%)	Nerastas (%)	Vidutinė pozicija (kai rastas)	Vid. gauto sąr. ilgis
125	12	13	15.7	59.3	3.01	3.46
250	22	18,1	25,5	34,4	2,99	5,7
500	30,9	20,9	31,4	16.8	3.17	8,49
1000	40,4	24,8	28,8	6	2,92	10,02
2000	47,2	23,8	26,5	2,5	2,62	12,29
4000	51,5	25,7	22,5	0,3	2,26	14,15

Lentelėje pateikiami eksperimentų rezultatai, kai kiekvieno ilgio fragmentams buvo atliekama po 10000 bandymų:

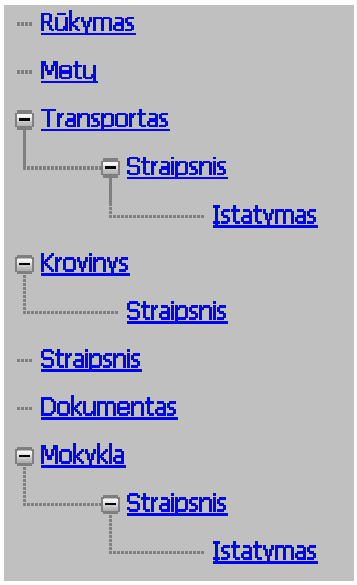
Fragmento ilgis	Sąraše pirmas (%)	Sąraše antras (%)	Rastas kitur (%)	Nerastas (%)	Vidutinė pozicija (kai rastas)	Vid. gauto sąr. ilgis
125	13,27	10,22	15,95	60,56	3,13	3,3
250	23,22	16,44	25,81	34,53	3,11	5,83
500	32,26	21,94	31,21	14,53	3,1	8,46
1000	38,81	24,23	30,79	6,17	2,89	10,21
2000	44,91	25,26	27,84	1,99	2,73	11,68
4000	51,89	26,54	21,37	0,2	2,26	13,76

Kaip matome, ir vienos ir kitos duomenų aibių atveju, eksperimentų atlikimo kiekis iš esmės neįtakojo gautų rezultatų, taigi galime laikyti, jog testavimo metodika teisinga. Išanalizavus pačius rezultatus galime padaryti tokias išvadas:

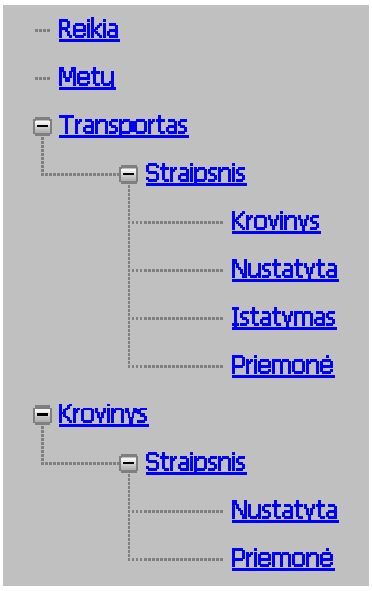
- Augant fragmento ilgiui, auga tikimybė rasti dokumentą sąrašo viršuje, mažėja tikimybė jo išvis nerasti, bei didėja susijusių dokumentų sąrašo ilgis.
- Dokumentų kiekis ir turinys mažai įtakoja palyginimų kokybę, nes nors ir pradinis duomenų kiekis abejose tyrimų aibėse skiriasi > 25%, rezultatai pakinta nežymiai.
- Esant fragmentui ilgesniam nei 500 simbolių (su tarpais, skyrybos ženklais, nereikšminiais žodžiais) tikimybė, jog dokumentas bus rastas sąrašo 1-2 vietose, viršija 50%.

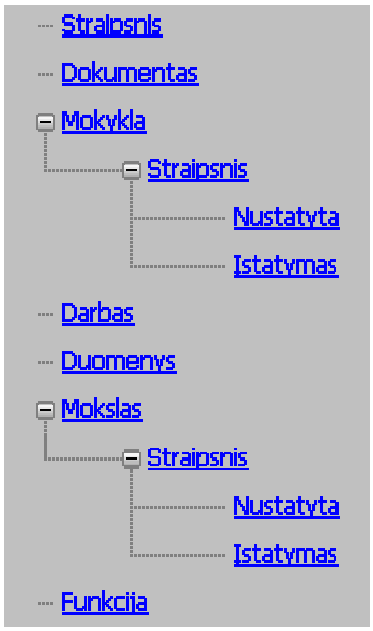
8.4 Sąvokų medžiai

Žemiau pateikiami sugeneruotų pagal 6.6 skyriuje aprašomą algoritmą sąvokų medžių vaizdai. Pav. 8.4.1 pavaizduotas pirmos duomenų aibės medis (žr. 8.3 skyrių), pav. 8.4.2 – antros duomenų aibės medis. Medžių generavimas buvo vykdomas su parametrais $j+1$ ir $r_{koef} = 0{,}5$.



8.4.1 pav. Sąvokų medis I





8.4.2 pav. Sąvokų medis II (perskirtas į dvi dalis)

9 Išvados

Pagrindinis klasifikavimo tikslas – efektyvesnė dokumentų paieška esant dideliems jų kiekiams. Paieškos efektyvumą apsprendžia paieškai reikalingi resursai (procesorinis laikas, saugomos informacijos kiekis) ir paieškos tikslumas (galutinis vartotojas randa dokumentą kurio ieškojo). Pabandysime tokiais kriterijais įvertinti šiame darbe pasiūlytus algoritmus ir metodus.

Populiariausias dokumentų paieškos būdas – pagal raktinius žodžius. Sintaksinės paieškos atveju, norint nustatyti kuriuose dokumentuose sutinkamas raktinis žodis, visų dokumentų visas turinys turi būti “skenuojamas”. Tam reikalinga labai daug procesorinio laiko, nors ir šiuolaikinės DBVS taikydamos įvairius metodus gali paiešką vykdyti itin efektyviai. Didesnis trūkumas – paieškos rezultatų kokybė. Į rezultatų sąrašą pateks ir dokumentai, kuriose raktiniai žodžiai nereikšmingi, o kirti dokumentai gali ir nepatekti, kadangi paieška bus vykdoma pagal vartotojo tiksliai užduotus žodžius ar frazę (pvz. žodžiai “transportas” ir “transportui” sintaksiniu požiūriu yra skirtingi). Taipogi sunku nustatyti, kaip teisingai surūšiuoti gautų rezultatų sąrašą (pritaikius sudėtingesnius metodus vėl augs paieškos laikas). Taigi abiejų efektyvumo kriterijų tokia paieška iš esmės netenkina.

Atlikus dokumentų apdorojimą šiame darbe aprašytais metodais gausime kokybiškai geresnę paiešką pagal raktinius žodžius: laiko atžvilgiu – reikės atlikti kur kas mažiau palyginimų (tik tarp dokumentą apibūdinančių sąvokų esančių kombinuotame grafe), efektyvumo atžvilgiu – paieška bus vykdoma pagal raktinių žodžių šaknis ir kamienus (paieška pagal sąvokas, o ne žodžius), rasti bus tik tie dokumentai, kuriuose raktiniai žodžiai reikšmingi, sąrašas bus surūšiuotas reikšmingumo mažėjimo tvarka. Tačiau čia slypi ir vienas didelis trūkumas: jei raktinis žodis nesutinamas kaip sąvoka grafe – dokumentai nebus rasti. Todėl galima teigti, kad **optimalius rezultatus vykdant paiešką pagal raktinius žodžius, galima gauti, kombinuojant šiame darbe aprašytus klasifikavimo metodus su sintaksine paieška**. Pvz. neradus rezultatų semantiniais metodais bandyti ieškoti sintaksiškai.

Kitas dokumentų paieškos būdas kuris dažnai pateikiamas vartotojams – kategorijų medis, kuriame sąvokos išdėstomos nuo pačių bendriausių medžio viršutiniuose lygiuose, iki specifiškiausių apatiniuose. Čia dokumentai yra susiejami su medžio viršūnėmis. Šio būdo privalumas yra tai, kad vartotojas gali matyti tam tikro medžio lygio kategorija ir daryti tinkamiausius pasirinkimus. Jei medyje nėra labai daug viršūnių, dokumentų paiešką galima vykdyti greitai, tačiau augant kategorijų skaičiui vartotojui

iškyla pavojus pasiklysti. Kitas didelis trūkumas yra tai, jog toks medis turi būti iš anksto paruoštas ekspertų ir dokumentai priskyrinėjami jame esančioms sąvokoms. Gali būti, jog naujam dokumentui tinkamos sąvokos medyje tiesiog nebus.

Todėl **šiame darbe pasiūlytas algoritmas, kaip automatiškai iš kombinuotojo grafo suformuoti kategorijų medį**. Tačiau būtina pastebėti, kad dėl semantinės informacijos stokos analizėje, tokį medį būtina pateikti ekspertų revizijai. Taipogi šį algoritmą galima būtų patobulinti (pvz. sudaranti sąvokų medžius visiems dokumentams, o vėliau juos integruojant).

Dar vienas aktualus paieškos būdas yra toks, kai pagal turimą dokumentą arba jo fragmentą bandoma rasti susijusius dokumentus (pvz. pagal aktualią įstatymo redakciją rasti anksčiau galiojusias redakcijas, įstatymus pakeičiančius duotąjį įstatymą ir pan.). Eksperimentų pagalba nustatyta, jog **remiantis pasiūlytais metodais, galima efektyviai (tiek mažomis resursų sąnaudomis, tiek paieškos rezultatų kokybės prasme) vykdyti susijusių dokumentų paiešką dokumentus arba jų fragmentus**. Pradinės eksperimentų sąlygos buvo suformuluotos taip, jog naudojantis sintaksiniais metodais, prasmingų paieškos rezultatų apskritai būtų neįmanoma tikėtis.

Būtina pasakyti, jog pasiūlytų metodų efektyvumo santykis su kitais klasifikavimo metodais yra sunkiai nusakomas dėl kelių priežasčių. Norint atlikti objektyvų palyginimą, reikėtų realizuoti kitus metodus ir bandyti atlikinėti neutraliai aprašytus eksperimentus su vienodais duomenims. Tačiau tokios užduotys liko už šio darbo ribų. Visgi atsižvelgiant į eksperimentų rezultatus, galima teigti, jog **aprašytieji algoritmai yra konkurencingi kitų klasifikavimo metodų atžvilgiu**.

Web-sąsajos realizacija parodo, jog **pasiūlytas klasifikavimo sprendimas gali būti panaudotas kuriant taikomąsias dokumentų apdorojimo sistemas**: tesės aktų, mokslinių darbų, informacinių pranešimų paieškoje, e-pašto šiukšlių (angl. *spam*) filtravime, pagalbos tarnybų (angl. *help desk*) sistemose ir kt. Net ir naudojant neoptimalius greičio prasme (pvz. XML) instrumentus, bei neoptimizavus išeities tekstų, dokumentų paieška vyksta pakankamai greitai, o papildomų duomenų saugojimas nereikalauja daug diskinės erdvės (~10-20KB dokumentui galutinėje XML struktūroje).

Žinoma, kad tolimesniais tyrimais galima būtų dar pagerinti aprašytų metodų klasifikavimo kokybę, pvz. dokumento analizės fazėje atpažįstant sąvokas sudarytas iš daugiau nei vieno žodžio, atsižvelgiant į jau sukauptus duomenis, integruojant vertinti konkrečių dokumentų įtaką bendrai klasifikavimo struktūrai.

10 Priedai

Prie darbo pridedamas kompaktinis diskas, kuriame yra įrašyti šis dokumentas, taipogi visi sukurti programiniai moduliai, jų išeities tekstai bei projektai. Disko turinys:

- Bylos „VBmag.doc“, „VBmag.pdf“ – magistrinio darbo dokumentai
- Katalogas „Source“ – programų išeities tekstai, projektai
- Katalogas „Programs“
    - Paruošti darbui moduliai „analyser1.exe“, „integrator1.exe“, „ControlUnit.exe“, pradinių leksinių duomenų byla „duomenys.xml“
    - Katalogas „Documents“ – analizei paruošti dokumentai
    - Katalogas „Input“ – katalogas analizuojamiems dokumentams
    - Katalogas „Output“ – katalogas integruojamiems dokumentams
    - Katalogas „Structure“ – katalogas integruotai duomenų struktūrai
        - Byla „structure_all.xml“ – visų testavimui naudotų dokumentų semantinis vaizdas
        - Byla „structure.xml“ – struktūros byla paruošta integravimui
- Katalogas „Boogle“
    - Paruošta darbui Web-sąsaja

Programų aprašymus ir darbo instrukcijas galima rasti 7.2, 7.3, 7.6, 7.7 skyriuose.

Literatūros sąrašas

[SJ03] Simon Jaillet, Maguelonne Teisseire, Jacques Cauche, Violaine Prince. Classification of Documents by Cotntents. Proceedings of the Second IEEE International Conference on Cognitive Informatics, 2003

[ZD02] Zhi-Hong Deng, Shi-Wei Tang, Dong-Quing Yang, Ming Zhang, Xiao-Bin Wu, Meng Yang. SRFW: a simple, fast and effective text classification algorithm. Proceedings of the first International Conference on Machine Learning and Cybernetics, 2002.

[TT02] Thorsten Teichert, Marc-Andre Mittermayer. Text Mining for Technology Monitoring. IEEE 2002.

[ZZ01] Zhou Zhi, Hinny Kong Pe Hin, Robert Kheng Lng Gay, Goh Wee Lin, Lee Shaur Yang. iTSum: One agent-based system for automated text summarizing. IEEE 2001.

[RL00] Renaud Leceuche. Finding comparatively important concepts between texts. IEEE 2000.

[BC00] B. Czejdot, J. Dinsmore, C. H. Hwang, R. Mille, M. Rusinkiewicz. Automatic Generation of Ontology Based Annotations in XML and their use in Retrieval Systems. IEEE 2000.

[WP01] Naïve Bayes Classifier. URL adresas http://en.wikipedia.org/wiki/Naive_Bayes_classifier. 117KB, 2005

[YY03] Ying Yang, Geoff I. Webb. On Why Discretitization Works for Naïve-Bayes Clasifiers. Proceedings of AI'03 LNAI, 2003.

[MS01] Manabu Sussano. Virtual Text Classification with Support Vector Machines. URL adresas: http://acl.ldc.upenn.edu/W/W03/W03-1027.pdf, 80 KB, 2003.

[AC03] Ana Cardoso-Cachpono, Arlindo Lmede Oliveira. An empirical Comparision of text categorzation methods. Url adresas: http://www.gia.ist.utl.pt/~acardoso/docs/2003-spire.pdf, 136KB, 2003.

[SK97] Sami Kasaki. Clustering methods. URL adresas: http://www.cis.hut.fi/~sami/thesis/node9.html, 11KB, 1997.

[RN03] Roberto Navigli and Paola Velardi, Aldo Gangemi. Ontology Learning and Its Application to Automated Terminology Translation. IEEE Inteligent Systems, 01/02 2003.

[LH03] Lars E. Holzman, Todd A. Fisher, Leon M. Galitsky, April Kontostathis, William M. Pottenger. A Software Infrastructure for Research in Textual Data

Mining. Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence, 2003

[RD03] Ruben Prieto-Diaz. A Faceted Approach to Building Ontologies. IEEE, 2003.

[AG03] Andrea Elaina Grimes and Robert P. Futrelle. Text Pattern Visualization for analysis of biology full text and captions. Proceedings of the Computational Systems Bioinformatics, 2003.

[AS01] Aixin Sun, Ee-Peng Lim. Hierarchical text classification and evauluation. 0-7695-1119-8/01 IEEE, 2001.

[AS04] Aixin Sun, Ee-Peng Lim, Wee-Keong Ng, Jaideep Srivastava. Blocking Reduction Strategies in Hierarchical Text Classification. IEEE transactions on knowledge and data engineering, vol. 16, NO. 10, 10/2004.

[MS98] Minoru Sasaki, Kenji Kita. Rule-based text categorization using hierarchical categories. 0-7803-4778-1 /98, IEEE, 1998.

[ID02] Inderjit S. Dhillon, Subramanyam Mallela, Rahul Kumar. Enhanced Word Clustering for Hierarchical Text Classification. ACM 1-58113-567-X/02/0007, 2002.

[NS89] N. Sližienė, A. Valeckienė. Lietuvių kalbos rašyba ir skyryba. Šviesa, 1989 m.

[AU06] Algimanto Urbanavičiaus svetainė. Lietuvių kalba ir literatūros istorija. URL adresas: http://ualgiman.dtiltas.lt