Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

FIIT-5220-29588

Bc. Dušan Zeleník

Tvorba odporúčaní využitím vzťahov podobnosti

Diplomová práca

Študijný program: Softvérové inžinierstvo
Študijný odbor: 9.2.5 Softvérové inžinierstvo
Miesto vypracovania: Ústav informatiky a softvérového inžinierstva. FIIT STU Bratislava
Vedúca práce: prof. Ing. Mária Bieliková, PhD.

máj 2010

ANOTÁCIA

Slovenská technická univerzita v Bratislave
FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ
Študijný program: Softvérové inžinierstvo

Autor: Bc. Dušan Zeleník
Diplomový práca : Tvorba odporúčaní využitím vzťahov podobnosti
Vedúca práce: prof. Ing. Mária Bieliková, PhD.
máj, 2010

Tento dokument opisuje postupy a výsledok práce v období začatom v septembri 2008 a ukončenom v máji 2010. Predmetom práce je problematika tvorby odporúčaní a jej úzke spojenie s hľadaním podobností, či už medzi používateľmi, alebo medzi odporúčanými entitami. V práci opisujeme využitie vzťahov podobnosti medzi entitami, pričom berieme v úvahu aj rozvíjajúce sa prostredie, v ktorom sa môžu nachádzať. Tvorbu odporúčaní založenú na obsahu sme postavili na efektívnej reprezentácií, pričom konkrétnu realizáciu demonštrujeme hlavne na textových dokumentoch a čitateľoch týchto dokumentov. Reprezentáciu a metódu odporúčania sme však navrhli tak, aby sme mohli pracovať s rôznymi typmi entít, ktoré je potrebné spracovať v reálnom čase. Vzťahy podobnosti a teda reprezentáciu vzťahov využívame pre generovanie obsahovo orientovaných odporúčaní, využitím pozorovania používateľa a jeho záujmov. Prácu sme vypracovali z pohľadu analýzy existujúcich riešení pre usporiadanie podobnosti a zjednodušenie vyhľadávania pomocou nej. Konkrétne návrhy riešenia a teda i samotného odporúčania využívaním podobnosti opisujeme v jednotlivých postupne nadväzujúcich častiach dokumentu.

Práca spočíva v aplikácii poznatkov študovaných metód a vytvorenia vlastnej reprezentácie, ktorá je spolu s metódou odporúčania overená pomocou implementácie v reálnom prostredí. Spracovali sme aktuálne materiály súvisiace s hľadaním podobností a tvorby zhlukov. Informácie o algoritmoch pracujúcich s priestorom vlastností objektov od algoritmov najbližších susedov až po komplexné riešenia tvorby zhlukov sú v dokumente taktiež zapracované.

Ďalším míľnikom je analýza problémov, ktoré súčasné riešenia majú a teda odhalenie miest, kde nie je možné použiť existujúce riešenia. Taktiež sme hľadali riešenia, ktoré je možné spojiť do jedného celku a tak poskytnúť nové významné výhody v jednom. Verifikáciu aktuálnosti vybranej problematiky sme vykonali popri štúdii existujúcich, nových riešení.

Z oblasti tvorby odporúčaní sme vybrali súvisiace projekty, ktorých riešenia nás inšpirovali. Analýza nedostatkov a nadobudnuté poznatky nás ďalej posúvajú k podrobnejšiemu opisu návrhu nášho riešenia. V práci postupne opisujeme metódy tvorby a modifikácie hierarchickej stromovej štruktúry, ktorá reprezentuje vzťahy podobnosti medzi článkami. Využitie takejto reprezentácie ďalej opisujeme z pohľadu výhod a riešení, ako napríklad udržovanie aktuálnosti článkov, alebo personalizované odporúčanie článkov. Tieto odporúčania sú založené na vlastnostiach entít a správaní sa jednotlivcov. Overenie reprezentácie podobností a metódy pre tvorbu odporúčaní je tak späté s reálnymi informáciami o používateľoch v doméne internetových novín.

ANNOTATION

Slovak University of Technology Bratislava
FACULTY OF INFORMATICS AND INFORMATION TECHNOLOGIES
Degree Course: Software Engineering

Author: Bc. Dušan Zeleník
Diploma project: Recommending based on similarity relations
Supervisor: prof. Ing. Mária Bieliková, PhD.
May, 2010

This document is on progress and results of the thesis done in the period started in September 2010. We focused on generating recommendations based on inter-entity similarity relations. We design a representation which works with entities which could be recommended or even with users who are consumers of these recommendations. In both cases we considered dynamics of domains which are evolving in the time. The method for recommending is based on efficiency achieved using our representation of similarity among text documents. Method for recommending is also ready to accept different entity types and works with these entities in real-time. Inter-entity similarity relations are then used to generate content based recommendations by monitoring the user and his preferences. Our work is also on analysis of existing solutions for entity classification and clustering and related similarity search. Our solution design and also recommender system are described in particular sections. Contributions are analysis of the related work and application of the knowledge to represent similarities. Similarity relations representation and recommender system is then evaluated in real environment.

We describe materials on discovering similarity and clustering which were processed besides methods working with multidimensional attributes of subjects. Complex methods are also explained in the scope of the thesis and their relevance to our solution.

Secondly, we explain, how these methods are insufficient in the dynamic environment. These facts have led to solution proposal and its implementation. Design of method for hierarchic representation of objects, especially text documents is explained and besides description of prototype implementation.

We also analyzed recommending systems, especially projects which inspired us. Drawbacks analysis and knowledge gained help us to provide more accurate design of our solution. In the document we mainly focus on four goals of the representation and method for recommending, which are dynamic adding of object to the clustered structure, providing creation of scalable clusters in time, solving content actualization and recommending using interest of users according to their personality. These recommendations are based on features of processed entities and behavior of users. Evaluation is finished with real information about real users in the chosen domain of internet news.

OBSAH

1 ÚVOD

Význam hľadania podobnosti medzi objektmi akéhokoľvek typu, či už je to jednoduchý text alebo komplexný videozáznam má zásadný význam pre usporiadanie informácií, čo umožní napríklad ich filtrovanie. Často sa tak dá zjednodušiť prístup k informáciám, ich vyhľadanie alebo napríklad aj predchádzať duplicitám. Z pohľadu dynamickej povahy takmer všetkých informácií (starnutie a pridávanie noviniek jednej témy, alebo i všeobecne), neuspeje žiadne riešenie manuálnej kategorizácie. Najmä z pohľadu používateľa vzniká automatizáciou lepší prehľad obsahu akejkoľvek množiny entít. Usporiadané informácie tak prinášajú jednoduchšiu navigáciu, a tiež vytvárajú prostredie pre ďalšie spracovanie aktivít používateľa a generovanie odporúčaní priamo na mieru špecifického človeka.

V problematike hľadania vzťahov medzi ľubovoľnými objektmi na základe ich podobností, či už sa jedná o texty alebo náročnejšie objekty ako obraz a zvuk, je už navrhnutých niekoľko funkčných metód [5, 29]. Ak však máme entít na spracovanie príliš veľa, súčasnými metódami nie je možné efektívne udržovať priestor, ktorý opisuje vzťahy medzi nimi. Neprítomnú dynamiku v podobe ťažkopádneho prijímania nových entít, alebo zoraďovania v čase, by pritom bolo vhodné vniesť v podobe okamžitého pridávania a odoberania entít, aby mala reprezentácia v každom momente ten najaktuálnejší obsah. Ak sa jedná o vopred neopísaný priestor vlastností entít, je zaujímavé hľadať dynamiku v rozširovaní tohto priestoru. Okrem dynamiky spojenej s modulárnosťou priestoru, ktorá sa neuvažuje pri bežných riešeniach objavujúcich vzťahy je zaujímavé hľadať aj dynamiku časovú. A teda sledovať vývoj v čase a poznať tak vzájomné vzťahy jednotlivých entít, ktoré sa priebežne menia.

Idea možného riešenia tohto problému nedostatku dynamiky vo vzťahoch medzi entitami je využitie princípov bežných zhlukovacích algoritmov [21] a ich modifikácií. Tie dokážu vytvoriť vhodné rozdelenie entít do skupín a tak zobraziť vzťahy medzi nimi (v skupinách majú silné vzájomné vzťahy). Problémom takýchto algoritmov ostáva ich časová náročnosť a teda nedostupná dynamika, pretože pri pridaní nového prvku sa tento môže iba odhadom vložiť medzi ostatné. Ďalším nedostatkom je neprekrývanie sa skupín vo viacerých z týchto algoritmov s výnimkou niektorých modifikácií [8]. Prekrývanie môže priniesť lepší pohľad na vzťahy medzi entitami a teda umožniť aj lepšiu dostupnosť dynamiky. Vzťahy môžu byť preto niekedy silnejšie a niekedy slabšie. Či už je to vzhľadom na dopyt hľadania vo vzťahoch (dôležitosť atribútov), alebo o zmeny v čase (vzory a periodicita slabnutia a zosilňovania vzťahov).

Ideálne je tieto vzťahy pri každej zmene znova získať, čo by však znamenalo opakované časovo náročné rozdelenie. Jedným z vhodných postupov môže byť udržovanie optimálnych veľkosti skupín a neustále preskupovanie týchto objektov tak, ako sa to využíva v peer-to-peer riešeniach [27] alebo v systémoch, kde treba rýchlo spracovať vstupy [7]. S novým objektom by sa tak spätne ovplyvnila len tá časť priestoru, ktorá je týmto objektom ovplyvniteľná. Časová náročnosť sa tým znižuje [28] a hľadanie vzťahov sa tak približuje k okamžitému dynamickému prístupu.

Generovanie odporúčaní sa môže uskutočniť v zásade dvoma spôsobmi. Buď hovoríme o odporúčaniach vytvorených pomocou sociálneho aspektu konzumentov odporúčaní, alebo využívame významy entít, ktoré sú odporúčané [13]. V prípade personalizácie odporúčaní môžeme oba spôsoby kombinovať, pričom sa zohľadňujú tak vlastnosti odporúčaných entít ako aj vlastnosti

používateľov. Hlavným problém je opäť efektivita algoritmov, ktoré pre hľadanie vzťahov medzi entitami vyžadujú už spomínanú ideálnu reprezentáciu.

Našou motiváciou je vytvorenie vhodnej reprezentácie priestoru entít a vzťahov medzi nimi, ktorý zahrnie aj možnosť úpravy vzťahov a ich rýchlu iteratívnu aktualizáciu. Zámerom tohto dokumentu je poskytnúť súhrn analýz existujúcich riešení súvisiacich s tvorbou odporúčaní, ale aj so spôsobmi reprezentácie vzťahov. Opisujeme problematiku a riešenia, ktoré následné využívame pri návrhu metódy hierarchickej reprezentácie vzťahov podobnosti. Táto reprezentácie je tak vhodná pre rýchle hľadanie podobných dokumentov (napríklad článkov v novinách) a tiež pre odhaľovanie záujmov používateľov. So získanými informáciami je možné v reálnom čase tvoriť odporúčania využitím vzťahov podobnosti odporúčaných entít.

Všeobecný prehľad metód zhlukovania dopomáha k uvedeniu aktuálnosti problematiky a nedostatky v súčasných riešeniach. V kapitole 2 uvádzame analýzu existujúcich metód, ktoré využívame pri návrhu nového riešenia. Keďže je nami navrhnutá metóda súčasťou rozsiahlejšieho projektu SME-FIIT[9], analyzujeme metódy postavené do prostredia spravodajstva opísaného v kapitole 3. Ďalej sa venujeme spravodajským systémom, ktoré v súčasnosti vlastným spôsobom vytvárajú odporúčania vo forme hľadania podobností a využívaním sociálneho aspektu čitateľov. Metóda okrem cieľov spojených s projektom v kapitole 4 nadobúda vlastnosti, ktoré sú využiteľné aj v ďalších doménach reprezentácie vzťahov medzi objektmi rôzneho typu (obraz, zvuk, alebo čokoľvek opísateľné množinou vlastností).

Návrh riešenia hierarchickej reprezentácie vzťahov je formálnejšie opísaný v kapitole 5 a jej využitie v tejto doméne uvádzame v kapitole 6. V tejto kapitole uvádzame návrhy riešení problémov domény internetových novín ako je zastarávanie článkov, odhaľovanie podobnosti článkov a stereotypov používateľov za účelom generovania odporúčaní článkov.

Riešenie a spôsoby overenia sú uvedené v kapitolách 7, 8, ktoré obsahujú špecifické poznatky nadobudnuté pri implementácií tohto riešenia a jeho ďalšej analýzy. Prílohy ďalej obsahujú technickú dokumentáciu v podobe diagramu prípadov použitia, diagramu tried a logického modelu. V prílohe tiež uvádzame vytvorené články súvisiace s predmetom práce a obsah digitálneho média, ktoré obsahuje elektronické časti dokumentu, ako aj úplnú implementáciu riešenia.

2 TVORBA SKUPÍN PODOBNÝCH OBJEKTOV

Analýza zhlukovacích algoritmov, ako prostriedku pre tvorbu dynamických skupín, má význam pre objavenie využiteľných vlastností, prípadne nedostatkov, ktoré zhlukovanie objektov prináša. Sústredením sa na podstatu týchto algoritmov umožňujeme hľadať štandardné už optimalizované riešenia, ktoré však napĺňajú ciele dynamického zoskupovania objektov. Od všeobecných princípov sa tak môžeme prepracovať až ku ideám, ktoré vznikajú využitím štandardných metód.

2.1 OBJEKTY V PRIESTORE

Ak sa pohybujeme v oblasti objektov, pod ktorými si možno predstaviť takmer čokoľvek z rôznych domén (kniha, film, obrázok), potom je vhodné reprezentovať ich v nejakej štruktúre. Jednoduchou predstavou môže byť napríklad matica, vyjadrujúca v stĺpcoch objekty a v riadkoch jednotlivé pojmy (tab. 1).

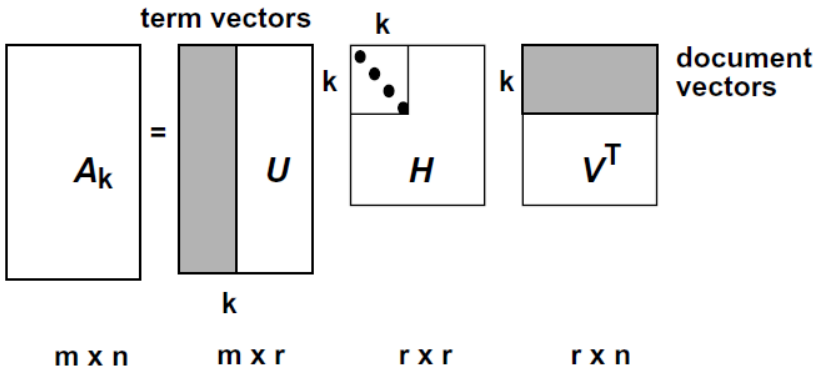
pojem	dokumenty								
	c1	c2	c3	c4	c5	m1	m2	m3	m4
človek	1	0	0	1	0	0	0	0	0
rozhranie	1	0	1	0	0	0	0	0	0
počítač	1	1	0	0	0	0	0	0	0
používateľ	0	1	0	0	0	0	0	0	0
systém	0	1	1	2	0	0	0	0	0
odpoveď	0	1	0	0	1	0	0	0	0
čas	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
dotazník	0	1	0	0	0	0	0	0	1
stromy	0	0	0	0	0	1	1	1	0
graf	0	0	0	0	0	0	1	1	0
minority	0	0	0	0	0	0	0	1	1

Tabuľka 1. Matica dokumentov a pojmov [14]. Vzťahy medzi dokumentmi a pojmami sú určené maticou obsahujúcou výskyty pojmov v jednotlivých dokumentoch

Atribút objektu (jeden z pojmov) je výhodné udržovať ako nezáporné reálne číslo, ktoré určuje silu tejto vlastnosti. Niektoré riešenia používajú aj triviálny model, ktorý je založený iba na prítomnosti, alebo neprítomnosti značky. Značka je v tomto prípade dvojhodnotový jednoduchý atribút. Objekt s atribútmi je zachytený spolu s ostatnými v jednej štruktúre. Takáto štruktúra umožňuje ďalšiu prácu s objektmi, ale aj ich vzťahmi. Častým problémom je miesto, ktoré matica zaberá v pamäti a teda aj réžia prístupu k jednotlivým poliam a práca s nimi. Vzhľadom na riedkosť matice, čo vlastne znamená prevyšujúci výskyt nulových hodnôt je zaujímavé redukovať tento priestor. Nulové hodnoty vznikajú v dôsledku rozličnosti jednotlivých objektov a teda nevyjadriteľnými hodnotami niektorých atribútov pre sémanticky odlišné objekty.

Tu nastupuje ľudská pamäť a teórie, ktoré hovoria o algoritmizácii schopnosti zapamätať si niečo. Zaujímavým je model holografickej pamäte [25]. Tento model pojednáva o ľudskej pamäti ako o distribuovanej reprezentácií informácií. Podporené je paradigmou neurónových sietí a časovej následnosti jednotlivých reprezentácií. Táto teória je taktiež postavená na pokusoch s léziami na mozgoch potkanov, ktorí boli schopní informácie v pamäti získať aj pri týchto poruchách. Informácia je teda v mozgu teoreticky uložená vo forme neznámej sémantickej reprezentácie. Ak máme teda množinu objektov, ktorú si chceme zapamätať, mozog hľadá koncepty týchto objektov (analógia synoným) a vytvára holografický model. Vektory pôvodných hodnôt atribútov pre objekty sa zobrazia na konceptuálne vektory. Interakciou konceptu, napríklad s narážkou na pamäť sa potom hľadá pôvodný objekt. Ide teda o formu redukcie celého priestoru vlastností objektov. Ak sa teda určí prah, pri ktorom možno odlíšiť pôvodné vektory príliš vysoko, človek by si pamätal veľmi málo objektov, ale veľmi presne. Ak sa prah znižuje, tak sa rozširuje aj pamäť, avšak možnosť napríklad hľadania asociáciami sa znižuje, lebo do jedného konceptu začnú patriť mnohé objekty. V zásade platí, že ľudský mozog má svoje ohraničenia, ktoré sú dané práve týmto modelom. Význam má teda hľadať optimálne rozloženie konceptov, napríklad odhaliť synonymickú podobnosť v texte.

V texte sa podobným spôsobom dá hľadať podobnosť jednotlivých slov [15]. Niektoré slová nesúce rovnaký význam by sa tak mohli normovať a zlúčiť. Tým by nevznikla stratová kompresia z pohľadu informačnej hodnoty, ale priestor by sa redukoval. Atribúty by niesli sémantickú informáciu. Tento problém okrem iného rieši dekompozícia matice vlastností [16, 19]. Dekompozícia je postavená na matematickom základe, kde je matica dekomponovaná na tri nové matice s novými vlastnosťami (obr. 1).

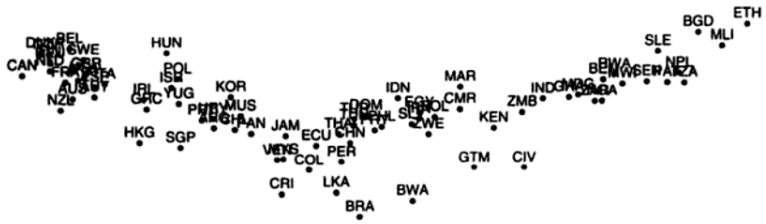


Obrázok 1. Dekomponovaná matica [16]. Neredukovaná matica A_k sa dekomponuje technikou SVD na dve ortogonálne matice (U a V) a diagonálnu maticu H. Vektory slov vystupujú ako koncepty, s redukciou nulového priestoru dokumentov. Naopak vektory dokumentov zohľadňujú redukciu nulového priestoru termínov.

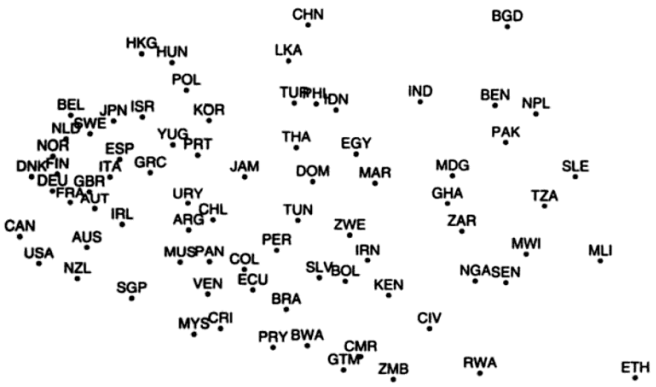
Dekompozíciou matice sa získa konceptuálny pohľad na dáta. Jednotlivé termíny a dokumenty tak vo vzájomnom vzťahu strácajú význam ako reálne termíny, alebo dokumenty. Vytvoria sa koncepty, ktoré sú redukciou zložitosti samotného priestoru termínov a dokumentov [5]. Technika matematickej dekompozície sa nazýva SVD (singular value decomposition) a často sa spája

s LSI (Latent Semantic Indexing). Toto indexovanie [14] využíva práve metódy dekompozície, aby sa tak odhalili vzťahy medzi objektmi na úrovni sémantiky. Metóda LSI vytvára sémantickú štruktúru textu. LSI využije SVD na redukciu priestoru [19], pričom sa vypustia sémanticky najnepodstatnejšie slová. LSI sa bežne využíva pri tvorbe odporúčaní, alebo hľadaní relevantných výsledkov pri vyhľadávaní v dokumentoch. Metóda je viazaná na text, ale jej idea je znovupoužiteľná v iných doménach. Táto idea je vlastne využitá pre vyhľadávanie nie na základe textu samotného, ale na základe konceptov, ktoré vytvára SVD. Koncepty teda sú virtuálne termíny, ktoré vznikli projekciou slovníka na redukovaný priestor. Eliminuje teda nedostatky bežného vyhľadávania, ako sú synonymá a podobne. SVD je okrem textu možné aplikovať úspešne aj na iné domény, preto je významným príspevkom pri hľadaní podobností a vzťahov.

Staršou alternatívou hľadania konceptov prostredníctvom LSI je využitie neurónovej siete. Kohonenové mapy [22] získavajú na vstupe jednotlivé objekty a ich atribúty. Vyvažovaním vzťahov sa neurónová sieť snaží vytvoriť mapu. Táto mapa premieta vzťahy na celý priestor, čím sa podobne ako pri LSI (projekciou) rozkladá priestor. Na obrázkoch (obr. 2, obr. 3) sú krajiny s použitím 39 dimenzií. Dimenzie charakterizujú vlastnosti krajín. Krajiny pritom môžu mať iné vlastnosti navzájom nekorešpondujúce. Výsledkom je organizovanie vzťahov a zviditeľnenie podobnosti jednotlivých krajín projekciou.



Obrázok 2. Neorganizovaná množina, priestorová projekcia [22]. Krajiny s 39 dimenziami sa projekciou zobrazia na dvojrozmerný priestor. Takáto redukcia priestoru je efektívna, avšak zanedbáva vzťahy medzi krajinami.



Obrázok 3. Organizovaná mapa krajín z obrázku 2 [22]. Všetky vzťahy sa ponechali, avšak zdôraznili sa minimálne rozdiely medzi krajinami. Obrázok tiež redukuje 39 dimenzií projekciou na dvoj rozmerný priestor. Použitím vzájomných podobností medzi krajinami je výsledok prehľadnejší.

Táto mapa je vlastne redukciou viacdimenzionálneho priestoru. Samotná mapa taktiež slúži viac iba na tieto účely priemetu, alebo projekcie inak nepredstaviteľného priestoru. Mapu môžeme pozorovať v 2D alebo 3D podobe. Je teda predstaviteľná aj pre človeka. Neurónová sieť sa však v tomto prípade učí neúmerne dlho. Ponúka statický výsledok, ktorý môže poslúžiť ako mapa konceptov. Tam sa potom môžu napríklad priraďovať nové objekty. Pridávanie však už je na základe rozdelenia počiatočných objektov, takže nie je možné s istotou nájsť rozdelenie bez opätovného preučenia sa mapy.

2.2 ZHLUKOVANIE A ZISŤOVANIE PODOBNOSTI

Tvorba zhlukov má svoju podstatu vo vytváraní skupín objektov, ktorých vzájomné vzťahy sú určené ich blízkosťou vo viacrozmernom priestore. Jednotlivé atribúty objektu tak tvoria hodnoty vektora, ktorý určuje bod, v ktorom sa objekt nachádza v N-rozmernom priestore. Umiestením objektu v priestore je potom možné hľadať najbližších susedov a odhadovať zhluky, v ktorých sa nachádzajú "blízke" objekty.

Algoritmus najbližších susedov vychádza z priestoru o rozmere rovnom počtu atribútov, ktorými jednotlivé objekty disponujú. V zásade je možné priestor transponovať a vytvoriť tak na osiach objekty a jednotlivé atribúty umiestniť do priestoru. V skutočnosti sa jedná iba o pootočenie matice, takže všetky vlastnosti ostávajú zachované. Algoritmus najbližších susedov teda využíva tento priestor a jednu z nasledujúcich možností výpočtu vzdialenosti.

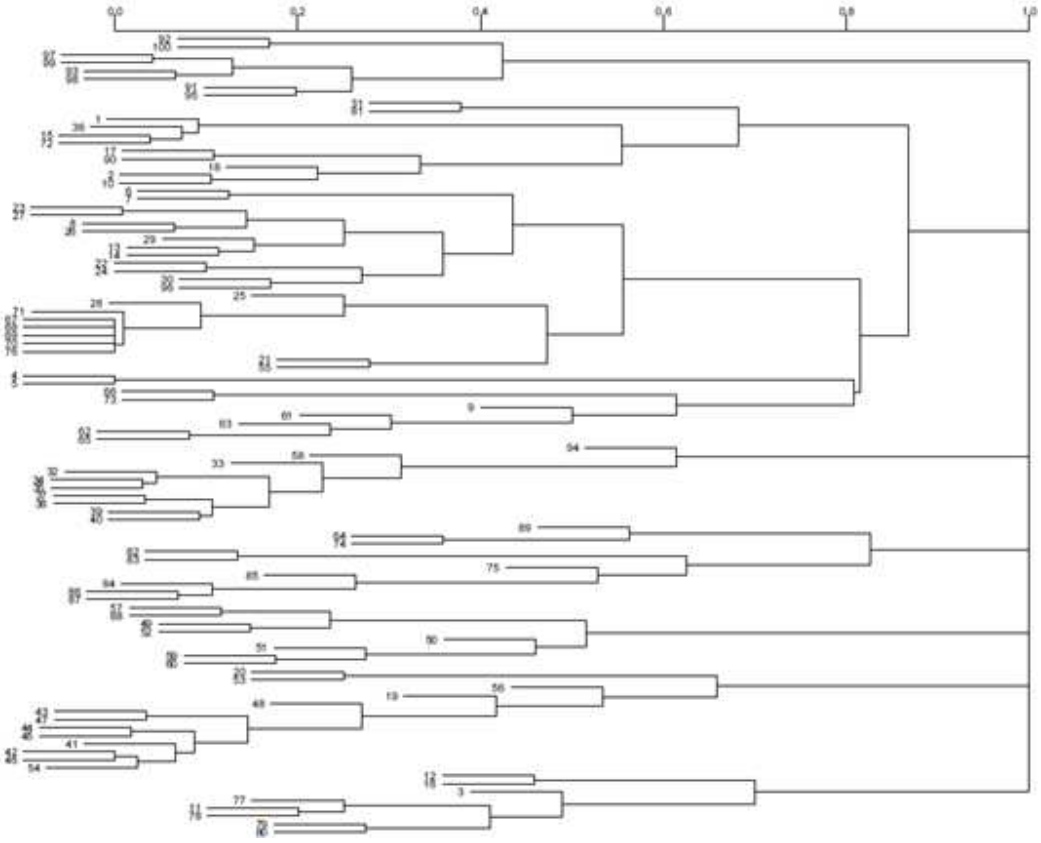
- $Cosine\ similarity = \frac{A.B}{\|A\|\|B\|}$
- $Euclidean\ distance = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2}$
- $Manhattan\ distance = \sum_{i=1}^{n}(p_i - q_i)$
- $Dice's\ similarity = \frac{2\,|X \cap Y|}{|X|+|Y|}$
- $Simple\ matching\ coefficient = počet\ spoločných\ atribútov$

V princípe sa jedná o výpočty vzdialeností na základe vektorov, ktoré určujú polohu prvku v priestore, alebo na základe inej metriky [3]. Výsledkom je vzdialenosť, ktorá sa potom používa na zoradenie objektov podľa ich blízkosti a teda nájdenie susedov. Ak si predstavíme priestor objektov ako vektory, potom zadaním jedného vektoru vieme nájsť napríklad jeho k – najbližších susedov.

Väčšina algoritmov vychádza z blízkosti susedov [4, 21] a teda parametrom na vstupe je zväčša statické číslo, alebo rozsah určujúci počet skupín, ktoré ma algoritmus odhaliť. Veľkosť jednotlivých skupín je potom priamo určená tým v akom sú objekty rozložení. Teda vznikajú aj také skupiny, ktoré majú veľmi nízky počet objektov a skupiny s vysokým počtom.

Pridaním možnosti prekrývania zhlukov [8] sa dá určiť aj príslušenstvo jedného objektu do viacerých skupín a teda využiteľne aj váha, s ktorou sa určuje príslušnosť jedného objektu vo všetkých zhlukoch. Prekrývanie zhlukov vyžaduje, aby každý zhluk poznal váhy objektov alebo opačne. Váha objektu a teda jeho miera príslušnosti k zhluku vyjadruje ďalšiu využiteľnú vlastnosť prostredia.

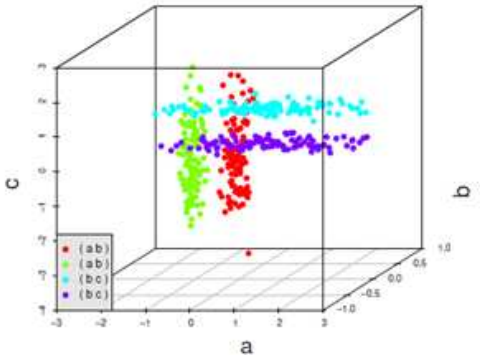
Ďalšou možnosťou ako vytvárať zhluky je využiť hierarchiu, ktorá môže vznikať pri ich tvorbe. Ak sú prvky na začiatku nezaradené, potom sa postupným aglomerovaním vytvárajú skupiny a tie sa opätovne zoskupujú do ďalších skupín (obr. 4). Vzniká tak strom, ktorý má výhodu, že vieme odrezať vetvu a znova usporiadať strom bez opätovnej nutnosti spúšťania celého algoritmu. Rovnako je jednoduchšia aj manipulácia so stromom pri pridávaní objektov alebo rozširovaní atribútov určujúcich ich podobnosť. Taktiež je výhodou možnosť zmeny počtu žiadaných zhlukov. Strom uchováva postupnosť krokov, ktorými boli zhluky vytvorené, a preto tieto zásahy nie sú časovo náročné. Nevýhodou hierarchického zhlukovania je nevyjadriteľnosť miery príslušnosti jednotlivých prvkov do zhlukov a teda nemožnosť dosiahnutia prekrývania zhlukov. Napriek tomu sa zo štruktúry dá určiť aká je váha príslušnosti k zhluku, v ktorom sa nachádza oproti ostatným prvkom.



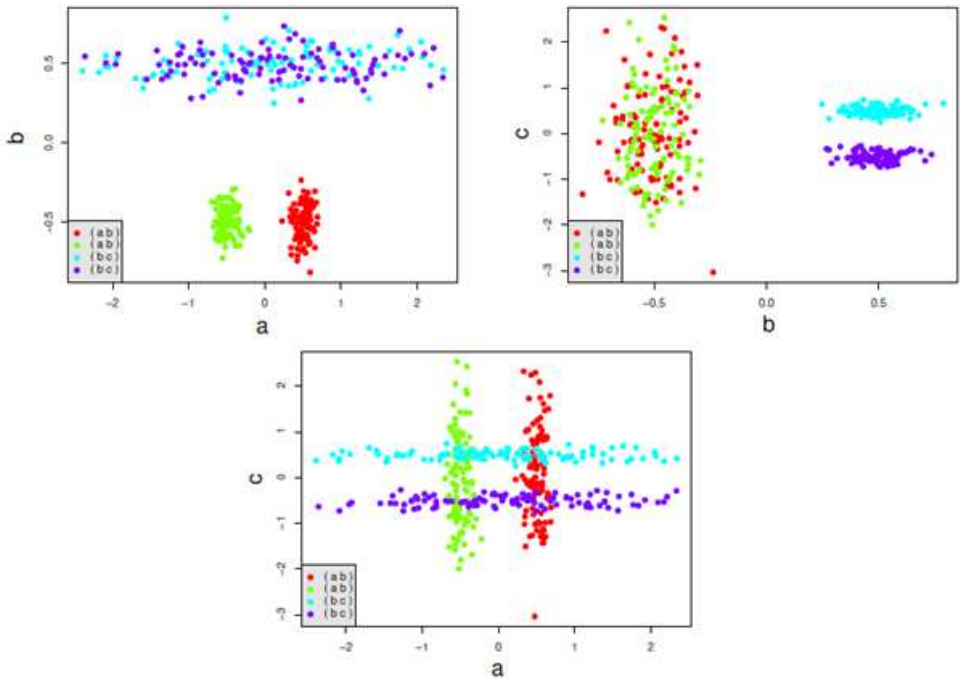
Obrázok 4. Príklad dendogramu pre 100 kníh [21]. Listy tohto stromu predstavujú jednotlivé knihy

Zaujímavé pohľady na zhlukovanie využitím podpriestorov sú cesty k výstupu smerom zhora nadol, alebo zdola nahor. Jednoducho možno označiť cestu zdola nahor ako postupnú projekciu N rozmerného priestoru na N-1 rozmerný priestor (obr. 5). Pričom sa využíva predpoklad, že projekciou sa viditeľnosť zhlukov nemôže stratiť, ale iba rozšíriť [4]. Čím samozrejme vznikajú problémy so zjednotením zhlukov. Tento postup však prináša pohľad na objekty a sledovanie ich vzťahov s meniacou sa váhou jednotlivých atribútov. Princípy zhlukovania zhora nadol sa vyznačujú aproximáciou výstupu, aby bol dosiahnutý najlepší výsledok. Pri tomto postupe je výhodné využiť

heuristiku, ktorá môže celú cestu hľadania výsledku skrátiť, ale hlavne ideálne nasmerovať. Postupy tohto druhu sú časovo náročnejšie, keďže sa algoritmus vykonáva pre celú množinu prvkov opätovne.



(a) Trojrozmerný a,b,c priestor s objektmi v zhlukoch



(b) Projekcia trojrozmerného priestoru a,b,c na kombinácie rozmerov z (a)

Obrázok 5. Projekcie priestoru [24]

2.3 METÓDY ZHLUKOVANIA

Zhlukovanie je spôsob ako spojiť podobné objekty a oddeliť od seba objekty rozdielne. Poznáme rôzne metódy zhlukovania [21]. Rozlišujú sa podľa spôsobov a priebehu hľadania konečné riešenia. Dôsledkom môže byť skutočnosť, že niektoré techniky sú vhodnejšie v špecifických prípadoch.

Výstupy zhlukovania môžu taktiež obsahovať špecifické informácie. Rozlíšme hlavné pohľady na tvorbu zhlukov:

- aglomeratívne a deliace,
- úplné a postupné,
- statické a fuzzy (dynamické),
- deterministické a stochastické,
- inkrementálne a neinkrementálne.

Aglomeratívne algoritmy začínajú s objektmi a postupne hľadajú cesty ako ich spájať do skupín. Hľadajú tak ideálne spojenia. Deliaci prístup začína akoby s jedným veľkým zhlukom, ktorý potom vhodne rozdeľuje na menšie časti.

Z hľadiska spracovania objektu je možné uvažovať ako do algoritmu vstupuje objekt. Ak sa postupne spracúvajú atribúty v nejakej následnosti, potom sa jedná o inkrementálne algoritmy s postupným rozvíjaním priestoru. Inou možnosťou je zoberať všetky atribúty objektov a na ich základe vytvoriť zhluky. Rozdiel spočíva najmä vo výhode dynamického rozširovania opisovaného priestoru inkrementálnymi algoritmami.

Statický prístup začleňuje jednotlivé objekty práve do jednej skupiny. Ak chceme umožniť, aby sa zhluky prekrývali, musíme pridať možnosť existencie prvku vo viacerých skupinách, a teda vyjadriť jeho členstvo v zhlukoch. Fuzzy prístup je využiteľný najmä ak sú prvky viac zviazané navzájom.

Každý algoritmus pre tvorbu zhlukov je optimalizačný. Ak sa algoritmus pokúša chybu redukovať na základe pravidiel, potom je deterministický. Príkladom stochastického prístupu môže byť napríklad hľadanie riešenia evolučnými algoritmami alebo neurónovou sieťou, kde hrá významnú úlohu náhoda.

Niekedy je vstupná množina príliš veľká. Pre redukciu času a pre získanie priebežných výsledkov vznikli najmä v minulosti algoritmy, ktoré dáta dávkujú, pričom udržujú stav zhlukov v ideálnom tvare. Takéto algoritmy sú označené za inkrementálne.

V dynamickom prostredí je vhodné uvažovať o využití práve tých spôsobov, ktoré sú schopné reagovať na meniace sa atribúty, neustále sa zvyšujúci počet prvkov, alebo spomínanú vzájomnú zviazanosť prvkov.

Ďalej opisujeme niektoré najznámejšie algoritmy [21] zhlukovania a ich prístup k tejto problematike. Prierez algoritmami sme zvolili, aby sme poukázali na vlastnosti rôznych typov algoritmov. Jednotlivé vlastnosti týchto algoritmov je možné meniť, kombinovať a rozširovať. Analýza týchto algoritmov pomôže získať všeobecný prehľad a následne predpoklady pre vytvorenie vlastnej metódy.

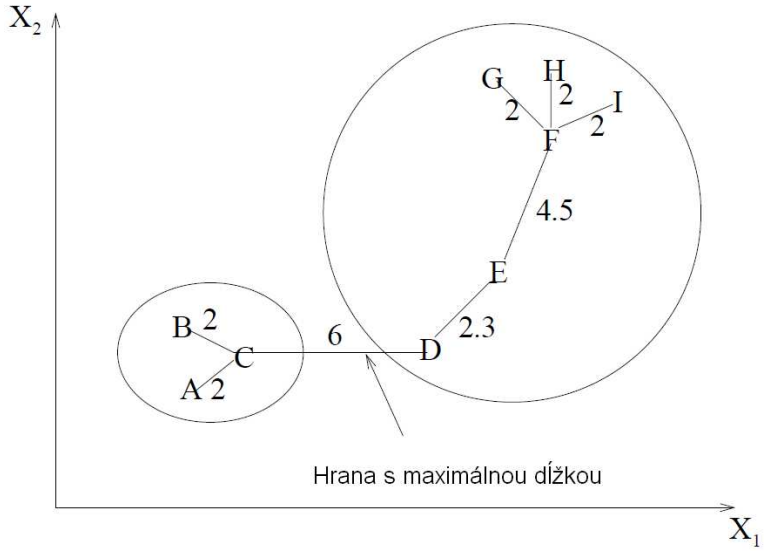
Leader-Follower. Tento algoritmus hrá významnú úlohu, ak sa jedná o dynamiku alebo časovú zložitosť. Patrí medzi inkrementálne algoritmy, ktoré vo svojej podstate pracujú s dátami postupne. Preto výsledky, ktoré dosiahne, nemusíme považovať za výsledné. Je možné pridávať nové prvky. Základom sú dve pravidlá. Je nutné poznať prahy pre určenie, kedy sú dva prvky blízke a kedy nie. Druhé pravidlo spočíva v posúvaní centra vzhľadom na meniace sa prvky v zhluku. Tento algoritmus

ako aj ostatné inkrementálne algoritmy sú ideálnym spôsobom ako udržiavať zhluky v meniacom sa prostredí, čo je dôsledkom možnosti priebežného pridávania nových prvkov.

K-means. Zrejme najznámejší algoritmus pre tvorbu zhlukov využíva zvolený parameter *k* na výber centier zhlukov. V prvom kroku sa tieto centrá náhodne vyberú z priestoru prvkov. Následne sa tieto všetky prvky priradia do najbližšieho zhluku. V ďalšom kroku sa centrá prepočítajú, aby sedeli so skutočným obsahom. Ak nie je splnené kritérium prekročenia štvorca chyby, tak potom sa celý algoritmus vykoná znova s výberom nových centier. Náhoda môže priniesť ideálne riešenie. Problém ostáva neustála nutnosť opakovaného spúšťania algoritmu a teda jeho časová zložitosť.

ISODATA. Tento algoritmus je obdobou k-means. Pre algoritmus k-means existuje množstvo obmien. Bežne sa mení prvý krok, ktorý náhodne hľadá správne centrá. Pridaním akejkoľvek heuristiky je možné zlepšiť základný algoritmus k-means. ISODATA vie pracovať už s vytvorenými zhlukmi. Oproti k-means je schopný odhaliť napríklad zlúčenie dvoch odlišných skupín. Ak je zhluk významne väčší ako ostatné zhluky, potom sa rozdelí napríklad na dva nové zhluky. Tento postup prináša dynamiku vo veľkostiach jednotlivých skupín. Dá prirovnať k optimalizovaniu ich veľkosti k ideálnemu počtu.

Shortest Spanning Path. Tento algoritmus má svoju podstatu v hľadaní miest pre rozdelenie. Vychádza z riešenia hľadania najkratších spojov medzi prvkami. Prvky sa v množine pospájajú tak, aby vytvárali minimálny strom. Potom je možné postupne zmazávať niektoré vetvy tak, aby sa vytvárali zhluky. Pritom prvé vetvy, ktoré sa zmažú budú tie najväčšie v strome (obr. 6). Toto riešenie je tiež inkrementálne. Poskytuje možnosť pridávať nové prvky do stromu počas jeho vytvárania. Prvky sa nespracúvajú naraz.



Obrázok 6. Odstránenie najväčšej vetvy a vytvorenie zhlukov [21]. Písmená reprezentujú dokumenty, čísla na hranách reprezentujú dĺžky. Prvý kandidát na rozdelenie je hrana s dĺžkou 6. Vznikajú dva zhluky. Celý priestor je v tomto prípade dvojrozmerný.

Single-line. Tento algoritmus patrí medzi aglomeratívne. Pracuje v troch základných krokoch. Najprv vytvorí toľko zhlukov, koľko je v skutočnosti prvkov v množine. Každý prvok sa vloží do jedného zhluku a vytvorí sa zoradený zoznam vzdialeností medzi všetkými zhlukmi. V ďalšom kroku sa opakuje spájanie prvkov do grafu tak, že spojenie sa vytvorí, ak sa nájdu najbližšie zhluky vzhľadom na prvky. Podmienkou ukončenia cyklu je spojitosť grafu. Posledným krokom je už samotná práca s grafom, ktorý tvoria vnorené zhluky. Tento graf je potom možné na rôznych úrovniach orezávať, čo umožňuje dynamický pohľad na veľkosti jednotlivých skupín.

Complete-line. Algoritmus patrí taktiež medzi aglomeratívne. Pracuje v troch základných krokoch. Najprv vytvorí toľko zhlukov, koľko je v skutočnosti prvkov v množine. Každý prvok sa vloží do jedného zhluku a vytvorí sa zoradený zoznam vzdialeností medzi všetkými zhlukmi. Tento algoritmus opäť opakuje spájanie prvkov do grafu tak, že spojenie sa vytvorí, ak sa nájdu také zhluky, ktorých spájací zhluk má minimálny priemer. Podmienkou ukončenia je však graf, ktorý je kompletne spojitý. Výsledkom je možnosť vytvárať rezy rovnako ako pri single-line. V tomto prípade však je možné dosiahnuť lepšie výsledky aj na dátach, ktoré sú vzájomne menej podobné.

2.4 DYNAMIKA ZHLUKOVANIA

Ak je snahou získavať obraz o zhlukoch priebežne a nie jednorázovo, môžeme hovoriť o dynamike. Aj u najjednoduchších algoritmov pre hľadanie zhlukov objektov je možné hľadať vylepšenia, ktoré s touto dynamikou pracujú. Na dynamiku zhlukovania vzhľadom na vzťahy objektov sa môžeme pozerať z viacerých pohľadov:

- pridávanie a odoberanie objektov (body v priestore),
- rozširovanie atribútov (rozmer priestoru),
- zmena vzťahov vzhľadom na meniacu sa prioritu atribútov (projekcie priestoru).

Ak uvažujeme pridávanie a odoberanie objektov je nutné vytvoriť zhluky tak, aby sa pri prípadnom pridaní, alebo odobraní množiny objektov znova dostalo prostredie do ideálneho stavu. Ak sa jedná napríklad o algoritmus k-means bolo by nutné opätovne spracovať celý vstup, aby bol výsledok hodnotný. V prípade zaradenia nových prvkov je možné vykonať vloženie už do existujúcich zhlukov, čo je často využívané, ale nepresné riešenie. Ak sa jedná o odstránenie prvkov, naopak môže nastať taký stav, v ktorom zhluk má napríklad nulový počet objektov. Zmena množiny objektov v prostredí zhlukov vytvorených napríklad algoritmom HACA [28] umožňuje lepšiu kontrolu nad jednotlivými objektmi a ich príslušnosť.

Ďalším prípadom dynamiky je rozširovanie prostredia o ďalšie atribúty objektov, napríklad pri pridávaní nového objektu. Doposiaľ sme nemuseli zvažovať atribúty, a preto sme niektoré napríklad pri redukcii rozmerov priestoru pre slová mohli zanedbať. V prípade zanedbania alebo redukcie atribútu nie je už návrat späť [2]. Riešenie sa nenachádza v zhlukovacích algoritmoch, ale v dekompozícii matice opisujúcej úplný priestor objektov. Dekompozícia SVD [16] sa úspešne používa pre zmiernenie rozmeru matice.

Dekompozícia upravuje maticu opisujúcu priestor v rámci možnosti bezstratovo. Teda redukcia spočíva vo vytvorení modelov so spoločnou sémantikou namiesto plného rozsahu atribútov. To znamená, že sa matematicky vytvoria mapy atribútov, ktoré majú podobný význam. Takáto dekompozícia sa využíva najmä v texte, avšak je možné si predstaviť aplikáciu na iné formy atribútov. Výhodou je dynamika, ktorú je možné čiastočne dosiahnuť a to zaradením nového atribútu do týchto máp. Ideálne, avšak časovo náročné by bolo spustenie dekompozície na celom priestore znova. Môžeme však predpokladať, že pri počiatočnom použití tejto dekompozície nedôjde k zanedbaniu žiadneho atribútu.

Posledným významným bodom je dynamika, ktorú možno dosiahnuť zmenou významu atribútov. Ak využijeme postupy podpriestorového získavania zhlukov, možno podobne meniť vlastnosti celého priestoru. Jednoduchými maticovými transformáciami tak možno zúžiť jednotlivé rozmery podľa ich priority. Takáto dynamika sa potom dá využiť nielen pri tvorbe dopytov získavanie blízkosti dvojíc objektov, ale aj na transformáciu prostredia vzhľadom na čas, ktorý periodicky, alebo podľa vzorov ovplyvňuje silu vzťahov medzi objektmi. Pre získanie transformačných matíc je nutné dlhodobejšie sledovať a získavať vzory opakovania sa napríklad záujmu o konkrétny filmový žáner počas týždňa.

3 VYUŽITIE PODOBNOSTI PRI ODPORÚČANÍ

Uplatnenie tvorby odporúčaní môžeme identifikovať v rôznych doménach. Zisťovanie podobnosti sa pritom často využíva ako riešenie pre generovanie odporúčaní. Tie môžu byť rôzneho charakteru, od najjednoduchších generátorov až po komplikované metódy. Bežne sa pritom odporúčacie systémy delia na dva typy a ich kombináciu:

- obsahové (content based),
- sociálne (collaborative filtering),
- hybridné.

Odporúčania kolaboratívneho typu, kde sa podľa skupiny používateľov generujú odporúčania na základe podobnosti týchto používateľov vyžadujú spracovanie tejto skupiny a ohodnotenie podobností. V prípade metód, využívajúcich vlastnosti odporúčaných entít je nutné taktiež vybudovať relácie podobnosti medzi týmito entitami. Obe metódy tvorby odporúčaní, ako aj ich možná kombinácia má spoločnú potrebu získavania podobnosti entít. Spracovanie, alebo zjednodušovania rozsiahlejšej množiny dát pre potreby odporúčania môže byť vykonané zhlukovaním, klasifikáciou, alebo inou formou reprezentácie podobností. Dôležité je zachovať efektivitu tejto metódy, aby odporúčania boli používateľovi poskytnuté čo najrýchlejšie (napríklad v prípade spravodajstva, kde sú odporúčané položky citlivé na čas).

3.1 METÓDY ODPORÚČANÍ

Odporúčania na základe obsahu. V odporúčacích systémoch založených na obsahu môžeme hľadať chyby ako napríklad ťažko spracovateľné entity, alebo rozpoznanie kvality. Predstavme si videá, ktoré by mohli byť týmito entitami. Video vieme v súčasnosti spracovať iba veľmi náročne. Môžeme vychádzať iba z kľúčových slov, alebo opisov videa. Inak sa k obsahu nedostaneme a nevieme určiť podobnosť dvoch videí. Napriek tomu môžeme predpokladať, že tento problém budeme schopní riešiť, alebo aj vieme, čo sa iných domén týka. Novinové články môžu byť jednoducho spracované, nakoľko sa jedná o jednoduchý text. Problémom však môže byť aj ako určiť kvalitu takéhoto článku. Predpokladajme, že používateľ má záujem vidieť ten lepšie napísaný článok s rovnakou témou. Reprezentácia podobnosti tak pri stránkach, s nekvalitným len ťažko pomôže takéto stránky odhaliť.

Ďalším významným problémom je nadmerná špecializácia (overspecialization). Tento problém vzniká v prípade obsahového odporúčania často. V prípade generovania odporúčaní z histórie a podobnosti obsahu, systém udržuje používateľa v uzavretom priestore odporúčaní. Na druhej strane, ak čitateľ novín prečítal jeden článok v danej špecifickej téme, potom pravdepodobne nemá záujem čítať články s tou istou tematikou. Avšak tiež možno predpokladať, že čitateľ má väčší záujem o túto tému, ako o tému, o ktorú sa nikdy nezaujímal. Odporúčacie systémy ako DailyLearner [10] dokonca filtrujú odporúčania príliš podobné histórií používateľa okrem

odporúčaní, ktoré s jeho históriou vôbec nesúvisia. V prípade, že používateľ je novým používateľom, ťažko z jeho histórie vygenerovať týmto spôsobom akékoľvek odporúčania.

Odporúčania na základe kolaborácie. Ak poznáme preferencie používateľa, alebo jeho vlastnosti, či históriu aktivít môžeme ho jednoducho zaradiť do prostredia medzi ostatných používateľov a hľadať prepojenia medzi nimi na základe ich podobnosti. Ak poznáme podobnosť medzi používateľmi, potom vieme podľa väčšiny podobných používateľov odporúčať. Tento spôsob odporúčania je podobný ako rady od priateľa, ktorý má pravdepodobne podobné záujmy. Riešenie týmto spôsobom však naráža na problém, keď do systému prichádza nová entita, napríklad nový článok. Tento článok nie je možné odporučiť, až kým ho neuvidí aspoň časť používateľovi podobných čitateľov. V zásade tak vieme čitateľovi zaručiť, že článok mu určite neutečie, ale je veľmi pravdepodobné, že sa k nemu dostane skôr ako ho vieme odporučiť. S týmto problémom sa spája aj efekt popularity entít. Niektoré články môžu byť natoľko populárne, že ich odporúčanie je takmer zaručené pre každého používateľa. Tieto odporúčania potom môžu byť nahradené jednoduchým riešením pre odporúčanie top najčítanejších článkov.

S predošlým problémom súvisí aj nezvládnutie tlaku používateľov (shilling attack), ktorý chcú presadiť svoje obľúbené, alebo dokonca vlastné príspevky. Používatelia umelo hlasujú, aby ovplyvnili rozhodovane odporúčacieho systému. Takéto prejavy je dôležité identifikovať a filtrovať.

Problém s novým používateľom platí aj pre tento typ odporúčacích systémov. Odporúčania sú takmer nemožné v prípade ak príde nový používateľ, ktorý nemá históriu, ktorá by pomohla k jeho začleneniu do odporúčacej skupiny.

Riešenie nazvané GroupLens [23, 26] sa pokúša oddeliť odporúčací systém od rôznych spravodajských systémov tak, aby predišli spomínaným problémom. Čím viac ľudí používa odporúčací systém tým viac možností vzniká. Ak by existoval iba jeden odporúčací systém, potom by sa mohli odporúčania generovať na rôznych stránkach a predísť tak napríklad problému s neznámym používateľom. Ďalšie problémy sa riešia spoločne v jednom systéme, preto môže byť shilling attack jednoduchšie odhalený. Prepojenie viacerých stránok potom funguje ako služba pre každého prevádzkovateľa spravodajských stránok, ale aj pre čitateľa.

Hybridné odporúčania. Takéto systémy sú upravené tak, aby využívali lepšie z vlastností dvoch predošlých typov a vyhýbali sa problémom, ktoré vznikajú [12]. Systémy tak možno rozdeliť podľa spôsobu, ktorým kombinujú princípy [1]:

- kombinácia nezávislých odporúčacích systémov
- pridanie charakteristík obsahových do kolaboratívnych
- pridanie charakteristík kolaboratívnych do obsahových
- vytvorenie zjednocujúceho modelu

Od jednoduchého kombinovania výsledkov dvoch riešení, až po vytváranie pravdepodobnostných modelov pre rozhodovanie sa o použitý stratégií odporúčania tak je možne kombinovať oba princípy. Výsledné systémy je vždy treba vytvoriť tak, aby boli pre jednotlivé prostredia vhodné a pre používateľa správne. V práci opisujúcej projekt YourNews [6] sa autori sústredili na odhalenie inak

transparentných postupov pre tvorbu odporúčaní. Čím komplexnejší je systém (hybridné riešenia) tým menej je pre používateľa takýto systém dôveryhodný. Ľudia potom systém nevyužívajú. YourNews dáva možnosť používateľovi upravovať objavené preferencie a tak mu okrem iného umožňuje aj zlepšenie samotných odporúčaní. Tento spôsob nie je automatizovateľný, pretože človek pozná svoje záujmy najlepšie. S tým však prichádzajú problémy, keď ľudia nemajú záujem zasahovať do systému, rovnako ako sú málo aktívni pri označovaní záujmu alebo nezáujmu o daný článok.

3.2 PODOBNOSŤ ČLÁNKOV A SPRAVODAJSTVO

Metódy zhlukovania vedia s určitou presnosťou dosiahnuť zobrazenie prvkov množiny na centrá zhlukov. Podľa rôznych metrík sa tak prvky formujú podľa podobnosti, podľa kritérií a parametrov danej metódy. Znalosť podobnosti môže byť silným marketingovým nástrojom. Metódy môžu usporiadať napríklad filmy, a následne ponúkať zákazníkom požičovne také tituly, ktoré sú v rovnakej skupine ako ním už videné. Podobným spôsobom vieme aj uľahčiť vyhľadávanie informácií. Zaradením možných výsledkov do zhlukov vieme dopytovať iba časť priestoru, spresnenú centrom zhluku.

Metódy tvorby zhlukov sú vo všeobecnosti univerzálne. Dôležité je spracovať prvky, s ktorými metódy pracujú a parametrizovateľná metóda vypočíta v nejakom čase výsledné rozdelenie podľa podobnosti. Preto je možné implementovať tieto metódy v rôznych oblastiach.

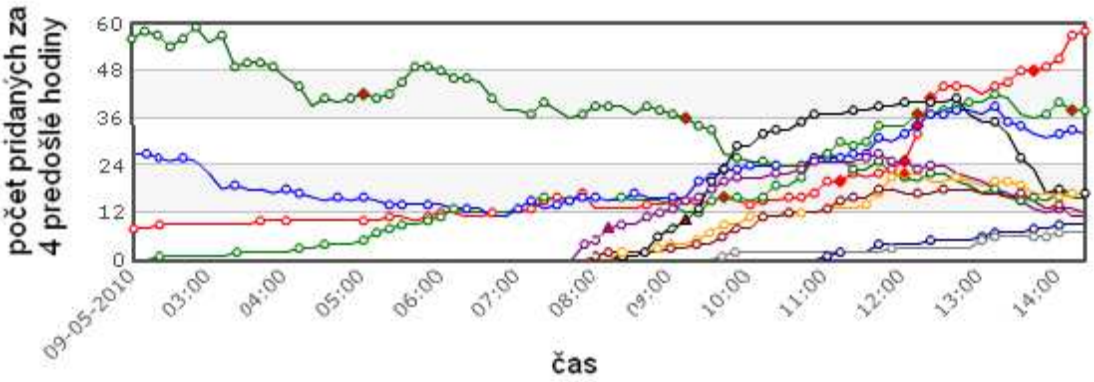
Špeciálnym prípadom je oblasť modelovania používateľa a využitie tvorby zhlukov na tvorbu skupín ľudí s rovnakými záujmami. Prirodzene sa tento proces deje v sociálnom prostredí, kde postupne ľudia určujú vzájomnú intenzitu vzťahov a teda podobnosť spoločných záujmov. V bežnom svete sa títo ľudia navzájom ovplyvňujú v rozhodovaní, z čoho vyplýva aj idea generovania odporúčaní v systémoch sledujúcich model používateľov. Akcie, ktoré sledovaný používateľ vykonáva sú zaraďované do skupín. Vzhľadom na vytvorené skupiny vie systém napríklad odporúčať knihu, ktorá zodpovedá jeho profilu. V tomto prípade je prvok množiny použitej v procese zhlukovania samotná črta človeka.

Hľadanie podobností informácií za účelom ich kategorizácie, alebo sprehľadnenia je v spravodajstve jednou z možností uplatnenia metód tvorby zhlukov. Zhlukovanie využíva metriky podobnosti napríklad pre články a ich atribúty slová. Čitateľ tak môže získať lepší prehľad o súvisiacich správach, bez nutnosti manuálnej kategorizácie. Internetové noviny často manuálne tvoria kategórie informácií, ktoré sprostredkúvajú. Tento úkon však nie je bezchybný a je časovo veľmi náročný, čomu napomáha aj vysoký počet kategórií a ich hierarchia. Používateľ môže články prehľadávať katalógovým spôsobom, pomocou stromu kategórií.

Manuálne zadávanie podobných článkov. Autor článku pri vytvorení príspevku manuálne určí, s ktorými článkami správa súvisí (príklad sme.sk). Tento úkon však je len málo efektívny. Väčšina článkov tak často nemá uvedené súvisiace články, alebo ich je iba malý počet. Používateľ tak v niektorých prípadoch má možnosť sledovať časť toho, čo už v danej téme bolo uverejnené.

Články v kauzách a témach. Táto metóda sprehľadnenia informácií je v držaní sa vo vymedzenom priestore káuz. Tento problém je veľmi náročne automatizovateľný, preto si ho môžu dovoliť skôr menšie internetové noviny venujúce sa špecifickým oblastiam (príklad tyzden.sk). Články nie sú opätovne usporadúvané automaticky, ale vymedzením káuz sa stáva hľadanie informácií jednoduchším pre čitateľa. Názov kauzy je tiež lepším ukazovateľom, ako je kategória, alebo centrum identifikované niektorou z metód tvorby zhlukov. Rozsiahlosť týchto novín však nepokrýva príliš široké spektrum spravodajských informácií. Ak by sa počet článkov zvýšil, znamenalo by to zvýšenie náročnosti triedenia článkov. V súčasnosti sú dokonca niektoré články, ktoré nie je možné zaradiť do káuz, čím sa vytráca univerzálnosť tohto riešenia.

Automatické porovnanie článkov. Existujú portály zhromažďujúce články rôznych novín vo viacerých jazykoch (príklad emm.newsbrief.eu). Toto riešenie je automatizované. Články sa sťahujú z najvýznamnejších internetových novín danej krajiny a sú porovnávané s článkami z histórie. Takto sa získavajú podobné články a čitateľ môže sledovať dianie vzhľadom na jeden článok a jeho obsah. Portál taktiež poskytuje prehľad štatistík. (obr. 7) Spôsoby odhaľovania podobnosti nie sú známe, pričom riešenie obsahuje aj chyby vo forme označenia nesúvisiacich článkov za podobné. Toto riešenie využíva textovú podobnosť a je úspešné skôr s textami písanými v angličtine. Riešenie neberie do úvahy sémantiku správ.



Obrázok 7. Štatistiky o počte článkov venujúcich sa rovnakej problematike počas jedného dňa. Os y určuje počet článkov pribudnutých v téme a os x určuje čas. Samotné čiary v grafe sú identifikované témy - množiny podobných článkov. Krúžky sú jedinečné články.

Odporúčanie článkov. Iným smerom vylepšenia prehľadnosti je generovanie odporúčaní čitateľom na základe ich záujmov. Jednoduchým, nie však najlepším riešením je udržovanie počtu prečítaní jednotlivých článkov a vytvorenie rebríčka. Zobrazením rebríčka používateľovi dávame odporúčanie, s veľkou pravdepodobnosťou vhodné. Problémom je, ak sa čitateľ má potrebu vrátiť k historickým informáciám. Články vtedy už nie sú čítané, a nie sú jednoduchým systémom odporúčané. Napriek tomu sú však relevantné pre používateľa. Tu môže pomôcť práve zisťovanie podobných článkov.

4 CIELE PRÁCE

Hlavné ciele tejto práce sme určili ako vzájomné nadväzujúce časti. V prvom rade je to reprezentácia vzťahov podobnosti, kde našim cieľom nie je spracovanie textu, ani výpočet samotnej podobnosti, ale je to usporiadania entít podľa vzájomných podobností do efektívnej štruktúry, ktorá nám umožňuje výhodne pracovať s týmito entitami. Umožní nám vyhľadávať podobné entity a v rastúcom, dynamickom prostredí.

Ďalšou metódou, ktorú uvádzame ako hlavný cieľ je využitie štruktúry, ktorá vznikla a teda generovanie odporúčaní založených na obsahu. Odporúčania pokrývajú preferencie používateľov, ktoré sa podľa predpokladov v čase menia. Keďže sme naše riešenie posadili do experimentálneho prostredia internetových novín, obe metódy sme sústredili na prácu s textom, ako novinovým článkom a používateľom ako čitateľom týchto novín.

4.1 VYTVORENIE METÓDY PRE EFEKTÍVNU REPREZENTÁCIU PODOBNOSTI TEXTOV

Používateľ portálu sme.sk má záujem vidieť články, ktoré súvisia s jeho záujmami. Prvým krokom a je teda vytvorenie postupu pre vyhľadanie najpodobnejších článkov ku konkrétnemu článku. Podmienkou je efektívna reprezentácia vzťahov obsahovej podobnosti, ktoré je možné objaviť medzi článkami. Efektivitu pritom uvažujeme najmä ako zložitosť vyhľadávania článkov v štruktúre a tvorba zoznamu najpodobnejších článkov.

Ak v prostredí internetových novín neuvažujeme používateľovu históriu čítania a jeho preferencie, tak tento spôsob môže byť forma odporúčania, alebo pomôcky. Súvisiace články pomocou tejto metódy vyhľadajú na základe článku, ktorý si čitateľ práve zobrazil a množiny jemu podobných článkov. Ak používateľ o článok a tému, na ktorú narazil nemá záujem, nemusí prejsť na súvisiace články. Ak však používateľ má záujem, je pre neho pripravených niekoľko článkov, ktoré sú mu najbližšie. Tento zoznam pritom obsahuje aj články, ktoré boli práve pridané. Takže používateľ získava najnovšie správy k problematike, ktorej sa práve venuje.

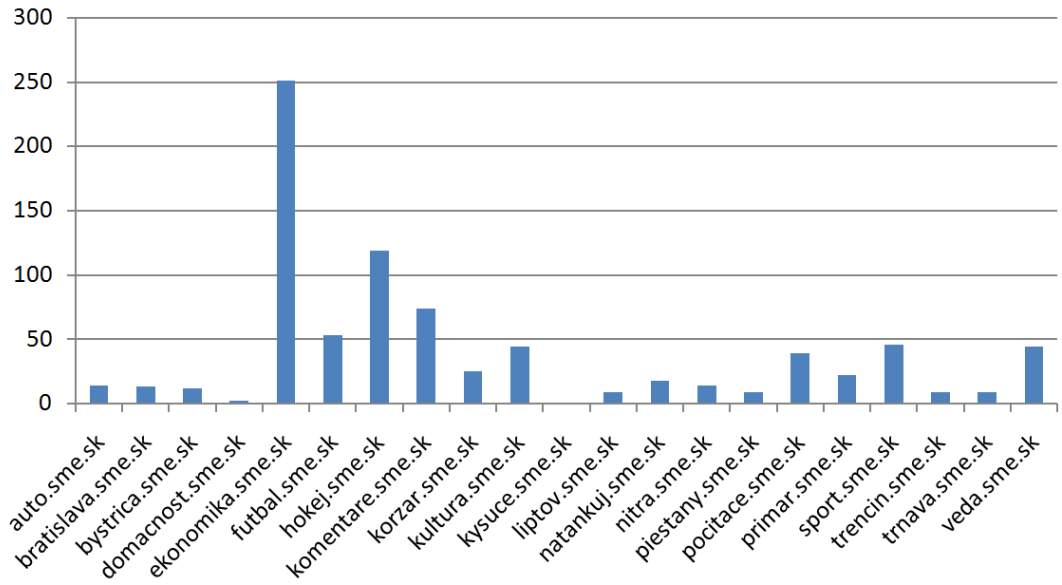
Podobnosť článkov je problém, ktorý riešia rôzne denníky. Výpočtom alebo manuálne možno získať podobné články, ktoré majú spoločnú tému alebo sú z rovnakej kauzy. Využiť možno napríklad kategórie článkov, ale obmedzuje sa tak podobnosť, ktorá môže nastať medzi článkami z rôznych kategórii.

Výpočet podobnosti článkov nie je našim cieľom, využívame existujúce riešenie pre odhalenie články obsahovo príbuzné. Naša metóda je schopná pracovať s rôznymi výpočtami a preto môže byť súčasný vzťah podobnosti nahradený iným vzťahom. Ak sa použije výpočet mieri príslušnosti článkov do rovnakej kauzy, potom sa vzťah zmení, ale reprezentácia tieto vzťahy zachytí. Našim cieľom je správna a efektívna reprezentácia, pomocou ktorej sme schopní usporadúvať články do skupín podľa určeného vzťahu. Pomocou takto reprezentovaných vzťahov sme ďalej schopní hľadať najintenzívnejšie prepojenia medzi vyšetrovaným článkom a ostatnými reprezentovanými článkami.

4.2 VYTVORENIE METÓDY PRE TVORBU ODPORÚČANÍ ZALOŽENÝCH NA PODOBNOSTI

Každý čitateľ má svoje záľuby, ktorých môže byť viac. Jednotliví čitatelia sú pritom odlišní, a preto by odporúčania článkov mali zohľadňovať osobnosť a jedinečnosť tohto čitateľa. Na rozdiel od hľadania podobných článkov k tým, ktoré používateľ práve číta, je ďalším cieľom hľadanie takých článkov, ktoré súvisia s jeho históriou.

História používateľa sa zaznamenáva spôsobom, ktorým nemožno spoľahlivo určiť, či používateľ naozaj prejavil záujem o zobrazený článok. Ak však berieme do úvahy, že jeho história aspoň čiastočne pokrýva jeho záujem, táto informácia sa ďalšou analýzou prejaví ako vystupujúci bod záujmov (obr. 8).



Obrázok 8. Zoznam záujmov náhodného používateľa. Čitateľ napriek vysokému počtu článkov jasne prejavuje záujem o vybrané témy. V tomto prípade sú uvažované iba najvyššie kategórie. Jednotlivé kategórie majú aj svoje podkategórie, manuálne vytvorené.

S vypustením informácie o kategórii článku, a zavedením informácie o podobnostiach článkov, možno čitateľovi priradiť záujmovú oblasť, čo naplní tento cieľ. Oporným bodom je jeho história, ktorej s rastúcim množstvom rastie aj kvalita možného odporúčania.

Keďže čitateľ sa mení, dynamika jeho záujmu sa v čase môže meniť taktiež. Najdôležitejším je pri tom záujem, ktorý sa prejavil naposledy, oproti záujmom z ďalekej histórie jeho správania sa. Princíp nie je založený na podobnosti jednotlivých ľudí, ale na predchádzajúcom správaní jednotlivého používateľa. Naplnenie tohto cieľa vyžaduje zvládnutie odhalenia podobnosti a efektívnu reprezentáciu týchto vzťahov. Hľadaním záujmových oblastí jednotlivého používateľa je možné odporúčať také články, ktoré spadajú pod takto určenú oblasť. Čitateľ tak dostane množinu článkov na mieru určených.

Problémom ostáva kvalita metódy, ktorá je schopná odhaľovať podobnosť článkov, a taktiež vo väčšej miere prítomnosť používateľa v špecifických záujmoch. S tým samozrejme súvisí aj kvalita

zaznamenávania jeho histórie, a schopnosť odhalenia skutočného záujmu o zobrazený článok. Našim cieľom je preto vyhľadávanie záujmov v reprezentácií, ktorá usporadúva články a podobnosti medzi nimi. Reprezentácia teda musí priamo umožňovať lokalizáciu záujmov a tým aj priamo podporovať tvorbu personalizovaných odporúčaní.

4.3 ZHRNUTIE

Hlavné ciele práce súvisia s vytvorením reprezentácie podobnosti a jej následné využitie pre odporúčanie článkov, ktoré sa reálne nasadí na spravodajský portál, kde bude aj overený. Hlavnými dvoma cieľmi sú:

- Návrh metódy pre efektívne porovnávanie textov
- Návrh metódy pre odporúčanie článkov založené na podobnosti

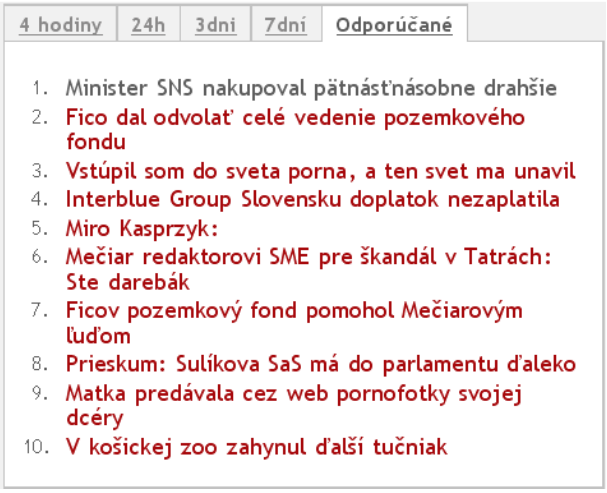
Keďže pracujeme s článkami, ktoré sú vo svojej podstate citlivé na čas a pribúdajú takmer neustále, musíme zvážiť aj časovú informáciu v procese hľadania podobných článkov a generovania odporúčaní. Popri hlavných cieľoch preto zvažujeme aj aktuálnosť článkov.

Témy starnú s článkami, ktoré ich tvoria. Ak teda skupina podobných článkov tvorí jednu tému, potom je jej aktuálnosť určená najnovšie pridaným článkom. Určovanie aktuálnosti má význam najme pri hľadaní podobných článkov - bez spojenia s históriou čitateľa. Môžeme totiž predpokladať záujem používateľa aj o menej aktuálne témy najmä ak nečíta noviny denne.

Proces hľadania podobných článkov, je postupnosť za sebou nasledujúcich rozhodnutí. Rozhoduje sa o väčšej podobnosti jednej z dvoch podmnožín článkov. S počtom rozhodnutí klesá úroveň abstrakcie, až sa dostávame na reálne - podobné články. Ak sa aj vyšetrovaný článok nachádza v systéme, jemu podobné články môžu byť výsledkom rozhodnutia na vyššej úrovni abstrakcie, v ktorej je už vhodné ďalšie rozhodnutia ovplyvniť časovou informáciou.

Množina článkov v systéme nesmie zastarávať, avšak musí poskytovať informáciu o vzťahoch medzi článkami. Pridávanie nových článkov preto musí byť priebežné a okamžite ovplyvniť ďalšie rozhodnutia o podobnosti. Takto sa dosiahne aktuálnosť obsahu a čitateľ hneď po pridaní nového článku môže byť k nemu nasmerovaný s vyššou pravdepodobnosťou, ako ku článku s rovnakou podobnosťou, ale starším dátumom. V skutočnosti sa jedná najmä o zoradenie výsledkov vzhľadom na atribút podobnosti a časovej informácie.

Portál sme.sk disponuje záložkou s možnými odporúčaniami a taktiež so zoznamom odkazov na podobné články pod samotným článkom. Riešenie odporúčania článkov je vhodné postupne (najprv menšej množine používateľov) zaradiť do záložky, ktorú môže čitateľ využiť. Táto záložka obsahuje zoradenú množinu odporúčaných článkov súvisiacich s jeho preferenciami (obr. 9). Podobne je vhodné riešiť aj momentálne slabšie zvládnuté riešenie podobných článkov pod článkom (obr. 10) a to nahradením manuálne vytvoreného zoznamu, zoznamom vygenerovaným našou metódou.



Obrázok 9. Možné rozšírenie záložiek o nový prvok. Nová záložka ponúka odporúčané články.

Obrázok 10. Súčasné riešenie zoznamu súvisiacich článkov pod článkom. Toto riešenie je manuálne, možnosťou a cieľom je jeho zámena za automaticky generované odkazy na podobné články v rôznych časových intervaloch.

5 HIERARCHICKÁ REPREZENTÁCIA VZŤAHOV PODOBNOSTI

Zvolili sme hierarchické usporiadanie podobností ako reprezentáciu, ktorá väčšiu vzorku dokumentov a vzťahov medzi nimi efektívne pripraví na ďalšie operácie. Takáto štruktúra je vhodná najmä v prostredí, kde je častejšou operáciou dopyt využívajúci vzťahy, ako operácie vkladania a odoberania. Hierarchická reprezentácia, ktorú uvádzame využíva postupne spájanie dokumentov podľa podobnosti, aby sme mohli na rôznych úrovniach podobnosti vyberať skupiny článkov a lokalizovať jednotlivé zhluky napríklad pri tvorbe odporúčaní.

V tejto časti sa postupne venujeme reprezentácií samotných dokumentov a následne prácu s reprezentáciou, v ktorej udržujme tieto články a podobnosti medzi nimi. Stromu, s ktorým pracujeme definujeme operácie, ktoré slúžia na postupné pridávane, výber a odstraňovanie článkov uložených v tejto štruktúre.

5.1 REPREZENTÁCIA MNOŽINY DOKUMENTOV

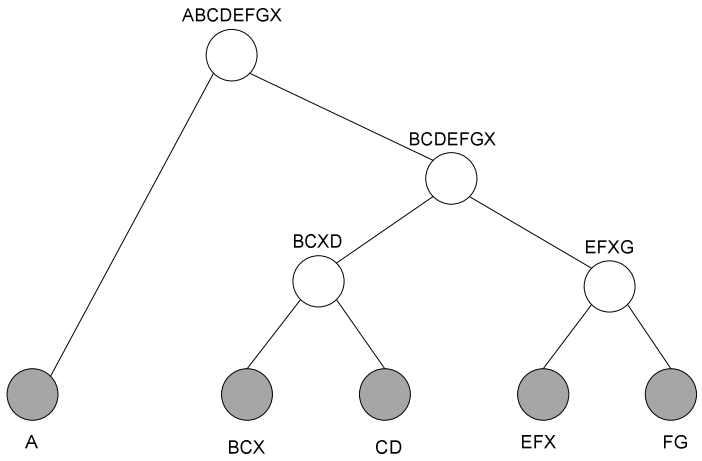
Navrhli sme reprezentáciu textových dokumentov, ktorá je potrebná pre výpočet podobnosti. Sústredili sme sa na efektívnu reprezentáciu, ktorou dosahujeme, že spracovaný vstupný text vystupuje v našom riešení ako objekt, s ktorým môžeme vykonať ďalšie operácie. Reálne dokumenty majú rôzne atribúty. V našom riešení pracujeme iba s textom samotným, aby sme sa vyhli ovplyvneniu podobnosti informáciami ako sú kategórie, sekcie a podobne. Vynechaním týchto atribútov sa sústredíme na podobnosť textov a teda aj dokumenty z rôznych domén môžu byť podobné.

Atribúty dokumentu sú v našom prípade hlavne slová, ktoré určujú vzťahy medzi dokumentmi, keďže výskyt rovnakých slov je predpoklad podobnosti dokumentov. Textové dokumenty sa dajú postupne spájať do dvojíc podľa podobnosti. Každú dvojicu reprezentujme ako nový dokument, ktorý však nepredstavuje reálny článok. Takýto dokument označujeme ako *metadokument*. Skutočný dokument, ako entita vystupujúca v našom riešení zahŕňa aj atribút času, ktorý označuje jeho aktuálnosť. Ďalej obsahuje vybrané slová a ich počty ako charakteristiky článku. Metadokument je pritom spojením skutočných dokumentov, a preto obsahuje spojenú informáciu o čase, slovách, a navyše informáciu o počte skutočných zjednotených potomkov.

Články, ale aj novovzniknuté metadokumenty postupne kategorizujeme. S využitím ich podobnosti vznikajú tak metadokumenty, ktoré opisujú množinu reálnych článkov. V prípade množiny slov, ktoré reprezentujú článok sa tento metadokument, ako zjednocujúci dokument, vytvorí zjednotením množín slov dvoch (podobných) dokumentov. Táto množina ďalej obsahuje iba unikátne slová, ktoré reprezentujú metadokument. Metadokument teda pôsobí navonok ako skutočný dokument, neviaže sa však na konkrétnu inštanciu reálneho dokumentu, ale na viaceré články. Keďže metadokument má vlastnosti obyčajného článku, je možné vytvárať metadokumenty aj spájaním metadokumentov navzájom, alebo tiež metadokumentu s reálnym článkom. Toto môže slúžiť pre hierarchickú klasifikáciu článkov. Formálne môžeme definovať graf ako strom, ktorý vzniká využitím vzťahov podobnosti. Platí nasledovné:

$Pre\ graf\ G = \{V, E\}, kde\ V$ sú $vrcholy,\ E$ sú $hrany$
$a\ pre\ V\ platí\ V = M \cup C, kde\ M$ sú $metadokumenty, C$ sú $reálne$ články
$a\ hrana\ e = \{v, v'\} je\ dvojica\ prepojených\ vrcholov,$
$pre\ \forall e, \forall v\ kde\ e \in E, v \in V\ a\ atr(v) je\ množina\ vlastností\ vrcholu\ v, platí,$
$že\ ak\ e_{ij} = \{v_i, v_j\}\ a\ e_{ik} = \{v_i, v_k\}, tak\ atr(v_i) = atr(v_j) \cup atr(v_k),$
$pričom\ platí, že\ |M| + 1 = |C|, |E| + 1 = |V|\ (tj.\ binárny\ strom)$

Metadokument obsahuje slová, ktoré má spoločné s článkami nižšie v hierarchii (obr.11). Okrem informácii o aktuálnosti podriadených prvkov, obsahuje aj informáciu o počte týchto prvkov. Táto informácia pomáha práve pri odhaľovaní úrovne (počet predkov), v ktorej sa metadokument nachádza a koľko úrovní zastrešuje (počet potomkov).



Obrázok 11. Strom reprezentuje reálne články (plné) a metadokumenty (prázdne). Písmená predstavujú zjednodušenú vlastnosť článku. Postupným spájaním vrcholov sa spájajú aj vlastnosti. Hrany predstavujú vzťahy medzi nadradeným a podradeným. BCXD je napríklad metadokument zastrešujúci reálne články BCX a CD. Pri metóde využitia množiny slov pre výpočet podobnosti článkov sa bude strom tvarovať podobne ako na obrázku.

Metadokumenty vznikajú aspoň v takom počte, v akom je udaný počet reálnych článkov, ktoré zastrešujú. Toto je dané najmä počtom priamych potomkov jedného metadokumentu. Ak je toto číslo nižšie počet metadokumentov sa zvyšuje. Články tak môžu byť v prípade párov potomkov zastrešené počtom metadokumentov rovnajúcich sa počtu reálnych článkov.

5.2 REPREZENTÁCIA VZŤAHOV PODOBNOSTI

Z hierarchického usporiadania reálnych článkov a metadokumentov, ktoré ich opisujú vznikajú vzájomné vzťahy, ktoré v našej reprezentácií vyjadrujú podobnosť článkov. Tieto vzťahy sa menia pri pridávaní nových článkov alebo odoberaní starých. Dynamický charakter týchto vzťahov musí byť preto rovnako dynamicky spracovaný. Vzťahy, ktoré reprezentujeme v našom riešení sa menia v závislosti od zmeny prostredia. Ak sa pridá nový článok, jeho vlastnosti dynamicky ovplyvnia vzťahy

tam, kde je to relevantné. Keďže sa dokumenty združujú podľa svojej vzájomnej podobnosti, tak sa ovplyvnia vzťahy iba medzi týmito dokumentmi.

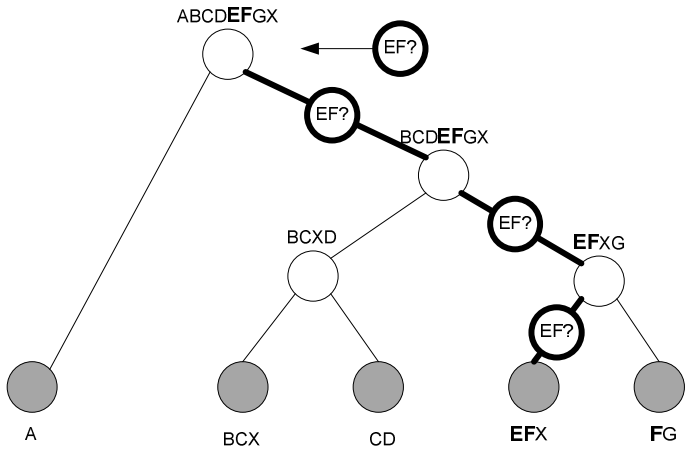
Štruktúra, ktorá vzniká využitím vlastností dokumentov je strom. Strom je v našom prípade binárny. Zvolili sme usporiadanie dokumentov do binárneho stromu, pretože sa tak znižuje počet potrebných porovnaní pri použití tejto údajovej štruktúry. To je v prípade odporúčania vo veľkých informačných priestoroch dôležité. V prípade stromu platí, že čím viac potomkov má jeden vrchol, tým viac porovnaní je nutné vykonať pri hľadaní podobných článkov, avšak závislosť je logaritmická.

Udržovanie párov potomkov tiež napĺňa predpoklad, že v každej množine existuje práve dvojica najpodobnejších prvkov. Čím bližšie sme k prvku, tým sa priebežne vypočítaná podobnosť zvyšuje. Ak podobnosť klesne, znamená to, že sme našli metadokument, ktorý obsahuje celú skupinu článkov, podobnú vyšetrovanému článku. Jeden z množiny reálnych článkov zahrnutých v takto určenej vetve je možno viac podobný k novému prvku, avšak tento vzťah neplatí opačne.

V prípade hľadania najbližšieho skutočného článku prebieha rozhodovanie na každom uzle až ku niektorému z listov stromu. Vyhľadanie metadokumentu, ktorý je najbližší k vyšetrovanému článku však prebieha takto:

```
def najdiNajblizsiMetaDokument(clanok)
      metadok = rootdokument
      while (podobnost(metadok,clanok) < starapodobnost)
            starapodobnost = podobnost(metadok, clanok)
            metadok = vyberBlizsiehoPotomka(metadok, clanok)
      end
      return metadok
end
```

Keďže pracujeme s binárnym stromom, znamená to prítomnosť práve dvoch priamych potomkov pre jeden metadokument. Reálne články ďalej nemajú potomkov, preto platí, že tieto vrcholy stromu sú práve aj listami. Ku konkrétnemu článku je teda možné postupne prejsť jednou cestou od koreňa stromu (metadokumentu zastrešujúceho celú množinu), cez metadokumenty, až po ukončenie vetvy stromu (reálny článok) (obr. 12).



Obrázok 12. Cesta k najpodobnejšiemu listu k článku EF. Podobnosť podľa vlastností.

Hrany stromu určujú, že prepojené vrcholy sú si podobné. Postupnosti hrán, ktoré je nutné prejsť ku reálnemu článku, určujú aj vzájomnú podobnosť jednotlivých reálnych článkov. Podobnosť cesty znamená podobnosť dokumentov. Od koreňa až ku článkom sa potom podobnosť určí na mieste, kde sa cesty rozchádzajú pretože platí:

$$Pre\ \forall v, v\ \in V, kde\ v\ je\ vrchol,\ V\ je\ množina\ vrcholov\ stromu,$$
$$pot(v) je\ množina\ všetkých\ potomkov\ vrcholu$$
$$a\ atr(v) je\ množina\ vlastností\ vrcholu\ v\ platí, že$$
$$atr(v) = \bigcup_{n=1}^{|pot(v)|} atr(pot(v)_n)$$

Miesto rozchodu je metadokument, ktorý určuje aj najnižšiu úroveň v hierarchii, ktorá zahŕňa vyšetrovaný článok. Metadokument zastrešuje celú množinu potomkov s vlastnosťami, ktorými aj sám disponuje. Takto sa v množine potomkov vyskytne reálny článok, ktorý je s uvažovaním danej metriky najpodobnejší k vyšetrovanému, ale aj ďalšie menej podobné články.

Reprezentovaním hierarchie sme vytvorili vzťahy nadradenosti a podradenosti v strome. Tento spôsob zohľadnil taktiež vzťahy, ktoré určujú podobnosťou jednotlivých dokumentov. Dva podobné dokumenty majú bližší spoločný nadradený prvok oproti dvom rozdielnym dokumentom.

5.3 TVORBA A MODIFIKÁCIA STROMU DOKUMENTOV

Hlavným princípom tvorby stromu dokumentov je využitie výpočtu podobnosti. Reprezentácia sa viaže na zvolenú metódu porovnania a výpočet podobnosti. Vytvorenie stromu priamo závisí od tejto metódy, preto nie je vhodné zmeniť ju po vygenerovaní časti stromu, čo môže ovplyvniť konzistenciu stromu. Rôzne inštancie stromu môžu využiť iné metódy porovnania alebo ich kombinácií. Kľúčovým je však poznať, aká je miera podobnosti medzi článkami. Pre výpočet podobnosti môžeme jednotlivé články vyjadriť vektormi, podľa slov v článku:

- vektor frekvencie výskytov slov z bázy (kosínusová podobnosť),
- vektor existencie slova z bázy (kosínusová podobnosť),
- vektor samotných slov článku (počet spoločných slov).

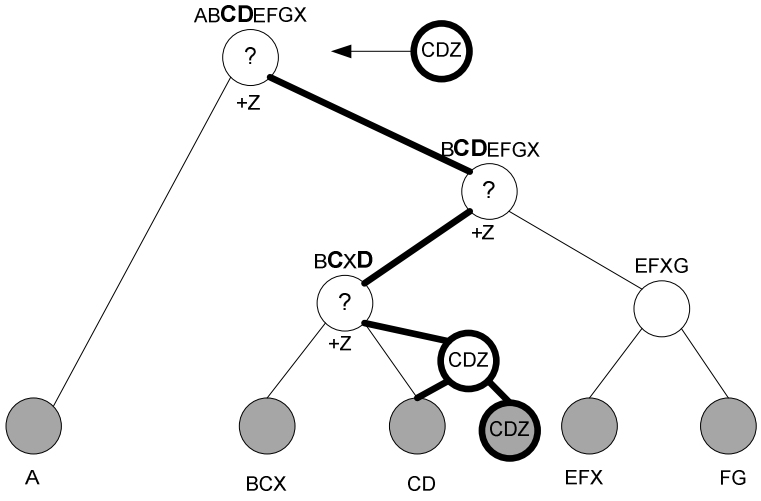
Výpočet podobnosti pritom považujme za atomickú operáciu, ktorá berie na vstupe dva dokumenty a vypočíta podobnosť medzi nimi vyjadrenú číselne v intervale (0, 1), kde 0 znamená žiadnu a 1 maximálnu podobnosť, zhodu. Okrem kosínusovej podobnosti sa javí ako vhodné využiť výpočtovo jednoduchšiu podobnosť podľa množiny spoločných slov. V našom riešení a hlavne pre overenie reprezentácie vzťahov sme využili nasledovný, zjednodušený výpočet pre určenie podobnosti dvoch článkov, reprezentovaných vektorom slov:

$$podobnosť = \frac{počet\ spoločných\ slov}{počet\ slov\ oboch\ dokumentov}$$

Reprezentácia, ktorú sme zvolili je efektívna najmä v redukcii počtu porovnaní, ktoré treba urobiť pri tvorbe a modifikácii stromu. Operácie využívajú predpoklad, že binárny strom postupne zjednodušuje vyšetrovanú množinu článkov. Ak hľadáme najpodobnejšie články k článku na vstupe, potom sa s každým porovnaním množina potenciálne podobných článkov zmenšuje. Ak by sme porovnávali vstupný článok so všetkými článkami v množine, bolo by to *N* operácií porovnania. Ak uvažujeme binárny strom, potom prebehne v ideálnom prípade $log_2 N$ operácií.

Nevýhodou je náročnejšie pridávanie a odoberanie prvkov, ktoré vyžaduje šírenie zmeny od miesta zmeny až po koreň stromu. Predpokladom je však, že zmeny nie sú tak frekventované ako dopyty, pričom zmeny možno vykonávať aj v dávkach. Čísla internetových novín [sme.sk] hovoria o priebežnom priemere 20 tisíc čitateľov oproti dennému priemeru 300 pridaných článkov. To znamená, že je potrebných podstatne menej modifikácií ako prístupov.

Strom sa vytvára postupným pridávaním prvkov do štruktúry. Prvok prechádza jednotlivými úrovňami v strome a rozhoduje sa o mieste, kde je vhodné sa pripojiť. Rozhodovací proces sa uskutočňuje na metadokumente a jeho výstupom je hrana napojená na potomka tohto dokumentu. Rozhodovanie súvisí s hľadaním bližšieho potomka a s podmienkou, že nesmie byť narušená silnejšia väzba potomkov. Ak teda vypočítaná podobnosť s potomkom klesne, našli sme najbližší metadokument, V prípade, že sa odhalí tento metadokument, prvok pridáme do jeho blízkosti (obr. 13), pričom blízkosť môžeme chápať rôzne v závislosti od vyžadovaných vlastností stromu. Blízkosť dvoch reálnych článkov je daná počtom prechodov cez metadokumenty na najkratšej ceste medzi nimi.



Obrázok 13. Pridávanie nového článku do štruktúry. Článok CDZ najprv ide cestou od koreňa stromu až ku najpodobnejšiemu vrcholu (v tomto prípade je to list). Pri každom prechode metadokumentom sa vykonáva porovnanie podobnosti s jeho potomkami. Ak sa lokalizuje miesto potom vznikne nový metadokument zastrešujúci nový pár reálnych dokumentov. Z obrázku vidno aj potrebu rozšíriť novú vlastnosť Z po tejto ceste až ku koreňu stromu.

Konkrétne miesto pridania môžeme uvažovať podľa charakteru dát. Rozdielom je pridávanie prvku v dvoch alternatívach:

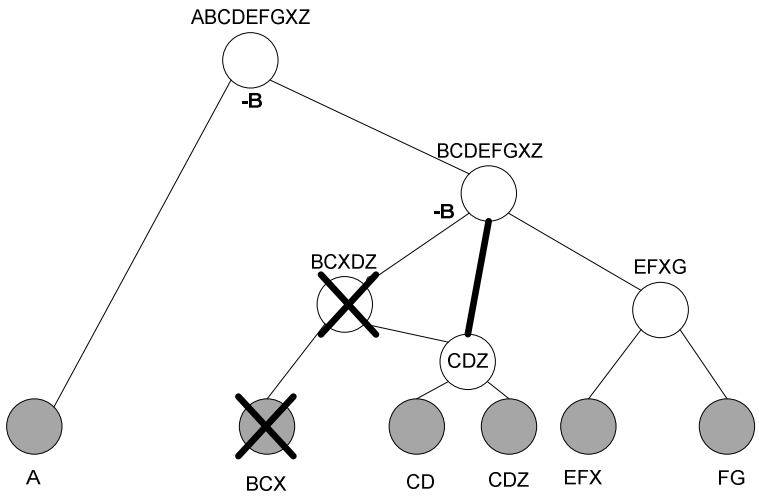
- rozvetvenie hrany medzi posledným akceptujúcim a jeho bližším potomkom (v prípade vysokej mieri rozličnosti článkov sa takýto strom stane vysoko spoľahlivý z pohľadu hľadania najpodobnejších článkov, ale taktiež veľmi hlboký, čo znižuje kvalitu vlastnosti prístupu a teda hľadanie je pomalé),
- rozvetvenie hrany za posledným akceptujúcim na mieste spresnenom podľa hĺbky stromu (tento prístup na úkor spoľahlivosti pridáva prvky tak, aby strom bol vyvážený - najširší a teda s vlastnosťou rýchleho prístupu).

Miesto pridania prvku môže byť jednoducho upravené vzhľadom na opatrenie pre vyhnutie sa príliš hlbokým, a teda neefektívnym formám stromu z pohľadu vyhľadávania. V takom prípade môže nastať rozvetvenie na najbližšej vetve, ktorá neprehlbuje strom. Podstatné je, aby sa prvok umiestnil na najbližšie možné miesto k vyhľadanému metadokumentu. Po pridaní prvku do stromu je nutné rozšíriť informáciu o jeho vlastnostiach po všetkých metadokumentoch na ceste ku koreňu stromu. Táto úprava mení vlastnosti týchto metadokumentov, aby boli ďalšie akcie vykonané nad korektným stromom.

Počet potomkov sa zvýši, pridajú sa slová a aktualizuje sa čas s informáciou o najnovšom potomkovi. Pridávanie prvkov uskutočňujeme nasledujúcim algoritmom pre vyváženie stromu:

```
def pridajClanok(clanok)
      metadok = najdiNajblizsiMetadokument(clanok)
      while (metadok.maPotomka? and metadok.hlbokystrom?)
            metadok = vyberMenejRozvetvenehoPotomka(metadok)
      end
      novymetadok = pridajMetadokumentNad(metadok)
      novymetadok.potomkovia = clanok, metadok
      novymetadok.rodic = metadok.rodic
      metadok = novymetadok
      while (metadok.nieje(rootmetadok))
            metadok.vlastnosti = metadok.potomkovia.vlastnosti
            metadok = medadok.rodic
      end
end
```

Proces odoberania dokumentov pracuje podobne ako pridávanie dokumentov, bez nutnosti hľadania miesta na pridanie. Prvok sa jednoducho lokalizuje a následne odstráni. Pri tomto akte zaniká aj metadokument a smerom ku koreňu stromu sa upravujú vlastnosti metadokumentov po ceste z tohto listu. Ostáva iba časť algoritmu šíriaca zmenu vlastností. Takto zabezpečujeme dynamickú reakciu na zmenu prostredia (obr. 14).



Obrázok 14. Pri odstraňovaní prvku sa ako prvý odstráni samotný článok. Následne sa musí odstrániť aj metadokument a upraví sa väzba druhého potomka, ktorý sa pripojí k rodičovi odstráneného metadokumentu. Tiež môžeme pozorovať rozšírenie informácie o odobratie vlastnosti článku BCX na ceste ku koreňu stromu. Odoberajú sa iba tie vlastnosti, ktoré nepokrývajú vlastnosti druhého potomka. V tomto prípade sa odoberá iba vlastnosť B.

Vychádzajúc z algoritmu pridávania nového článku, odoberanie prebieha ako odstránenie článku alebo celej vetvy a šírenie zmeny, ktorá pri odstránení nastala:

```
def odoberClanok(clanok)
      metadok = clanok.rodic
      metadok.rodic.potomok = metadok.potomok
      vlastnosti = clanok.vlastnosti
      while (metadok.nieje(rootmetadok))
            metadok = medadok.rodic
            metadok.vlastnosti = metadok.potomkovia.vlastnosti
      end
end
```

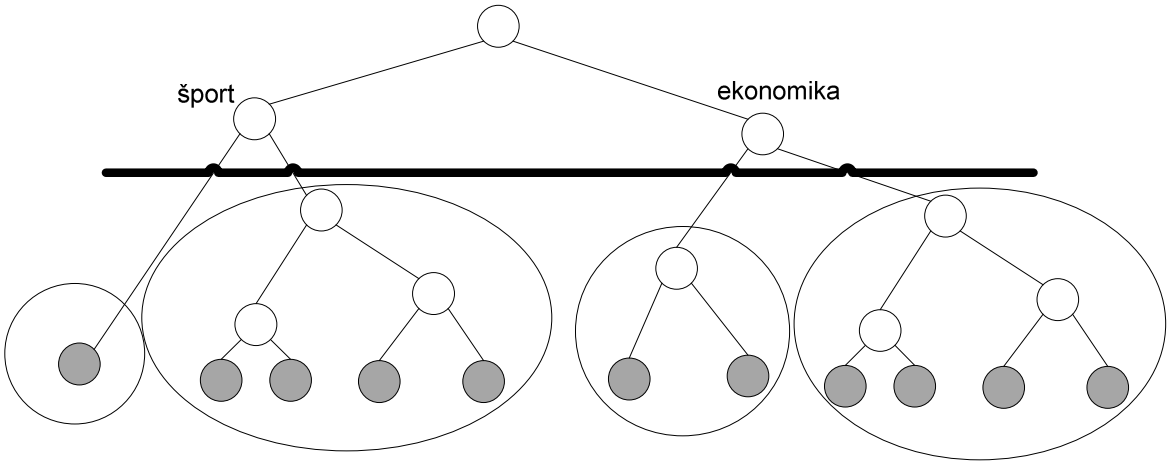
6 VYUŽITIE HIERARCHIE A DYNAMIKY

Hierarchia článkov vytvorená podľa vzťahov podobnosti prináša výhody, ktoré postupne využívame na napĺňanie stanovených cieľov. Štruktúra, ktorá vzniká postupným pridávaním článkom je využiteľná pre vyhľadávanie podobných článkov a tak umožňuje lokalizovať záujmy používateľov, na rôznych úrovniach. V tejto časti sa postupne venujeme tvorbe týchto skupín a následné generovanie odporúčaní s atribútom času.

6.1 TVORBA SKUPÍN PODOBNÝCH ČLÁNKOV

Jedným z prípadov použitia vytvorenej hierarchie je hľadanie podobných článkov. Toto hľadanie napĺňa nepersonalizovanú potrebu používateľov hľadať články podobné k práve čítanému článku. V tomto prípade sa jedná o formu odporúčania spojenú s používateľom iba podľa posledného navštíveného článku. Jeho záujmy sa tak dlhodobo nesledujú. Takéto odporúčanie môžeme vykonať ku akémukoľvek článku, avšak výsledok je obmedzený na prvky, ktoré sa nachádzajú v strome.

Články sa prirodzene pridávajú do hierarchickej štruktúry (binárneho stromu) podľa ich vlastností. Dôsledkom procesu pridávania na základe hľadania najbližšieho už pridaného prvku je štruktúra, v ktorej vznikajú skupiny podobných článkov. Tieto skupiny môžu byť vytvorené na rôznych úrovniach podľa metadokumentu, ktorý sa zvolí [30]. Metadokumenty tak zjednodušujú tvorbu skupín podobných článkov (obr. 15).



Obrázok 15. Rez stromom cez niektorú úroveň vytvára zhluky podobných článkov na nižšej ako kategorickej úrovni (kategórie sa nestotožňujú s reálnymi, uvedené sú orientačné názvy). Posúvaním tohto rezu sa mení jemnosť skupín.

Samotné generovanie skupín nemá význam, dôležitá je však táto vlastnosť metadokumentov z pohľadu hľadania "najbližších" článkov. Zadaním článku (aj v štruktúre nezapísaného), vieme vyhľadať metadokument, ktorý je danému článku najpodobnejší. Tento metadokument zastrešuje

skutočné články, ktoré sú mu najpodobnejšie. Takto získame presnejšiu množinu podobných článkv ako výsledok hľadania. Tie môžu byť predmetom hľadania ďalšej zjemnenej podobnosti a teda zoradené. Rozdiel je v získaní výrazne menšej podmnožiny článkov z celého informačného priestoru a jej následné menej náročné usporiadanie. Strom má informáciu o tom, akým spôsobom je možné vytvoriť skupinu podobných entít. Oproti statickým formám zhlukovacích algoritmov prináša túto formu škálovateľnosti na požiadanie.

Tvorba skupín a vôbec tvorba štruktúry závisí od metódy vyhodnocujúcej vzájomnú podobnosť dvoch článkov. V našej práci sa samotnému výpočtu podobnosti nevenujme, ale pre potreby experimentov uvádzame postupy, ktoré sme zvolili pri predspracovaní článkov.

Podobnosť je vhodné vypočítať na základe celého obsahu článku. Metódy ohodnocujúce podobnosť na základe nadpisov, alebo kratších začiatočných častí textu môžu dosahovať dostačujúce výsledky, avšak ide hlavne o zjednodušenie problému na úkor obsahu textu. Lepším spôsobom je redukovať vysoký počet dimenzií takéhoto článku. Naše riešenie hľadá podobnosť vektorov slov, ktorých zložitosť je redukovaná slovníkom.

- lematizácia (základné tvary slov)
- stemovanie (korene slov)

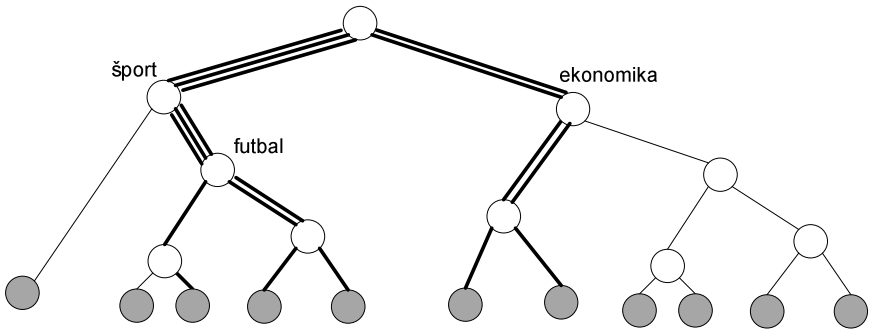
Predpoklad je, že táto redukcia neuberá z výpovednej hodnoty celého článku, redukuje iba variácie jazyka. Ďalšia redukcia spočíva v odstránení príliš opakujúcich sa slov (stop slová) a slov, ktoré sú naopak veľmi jedinečné.

6.2 ODPORÚČANIE ČITATEĽOVI

Oproti odporúčaniam v podobe tvorby skupiny podobných článkov k práve čítanému článku, ktoré nezohľadňujú dlhodobý záujem používateľa, vzniká aj potreba generovať odporúčania na základe modelu používateľa. Tento prípad dlhodobejšieho sledovania článkov, ktoré číta je jedným z využití hierarchickej štruktúry. S pridávaním článkov sa dynamicky mení aj množina používateľom prečítaných článkov.

Pridávanie článkov do štruktúry nie je orientované na model používateľa. Strom nepozná žiadne preferencie používateľa. Strom však disponuje metadokumentmi, ktoré na rôznych úrovniach určujú podobné oblasti záujmu. Predpokladom je, že záujmy používateľa a teda jeho model sa dá mapovať na oblasti určené týmito metadokumentmi. Metadokument v tomto prípade reprezentuje stereotyp záujmu. Každý čitateľ svojím postupným čítaním článkov zanecháva stopu a to v podobe článku, ktorý prečítal. Všetky články, ktoré používateľ prečíta sa potom stretávajú vzájomne v strome na týchto metadokumentoch (obr. 16). Podľa počtu prečítaných článkov sa potom určí hĺbka úrovne, na ktorej sa hľadajú stereotypy záujmov. Odporúčanie v tomto prípade pracuje ako sledovanie pridávania článkov a spustenie procesov pri prechode týmto metadokumentom. Ak sa vloží nový článok pod úroveň záujmovej oblasti čitateľa, dostane odporúčanie pri nasledujúcej návšteve v podobe zoznamu článkov. Okrem tohto zoznamu je možnosť odporúčať všetky články, ktoré spadajú do jeho záujmových oblastí a ešte ním neboli prečítané.

Takéto odporúčanie nepracuje so statickými modelmi používateľov. Rovnako ako sa menia metadokumenty a články v strome, alebo aj záujem používateľa, tak sa menia odporúčania, ktoré sú generované. Odporúčanie je pritom zamerané na obsah článkov a záujmy čitateľa. Odporúčania sú viazané na históriu jednotlivcov a model vyjadrený grafom používateľom prečítaných článkov.



Obrázok 16. Každá hrubá čiara predstavuje jednu cestu ku článku, ktorý si zobrazil jeden a ten istý používateľ. V tomto prípade sa používateľ zaujíma o futbal a ekonomiku. Kategórie v strome sa pritom nestotožňujú s kategóriami v realite (názvy sú uvedené iba pre ozrejmenie). Sú to virtuálne stereotypy záujmov. Takýmto označením potom vieme získať záujmové oblasti jednotlivých ľudí a napríklad odporučiť posledný článok z oblasti futbal, ktorý podľa obrázka čitateľ nevidel.

V strome vieme lokalizovať tie vrcholy, ktoré reprezentujú záujmy čitateľa. Čím viac sa čitateľ sústredí na konkrétnu tému, tým viac je jeho záujem špecifický. Ak z jeho histórie nevyplývajú silné vyhrotenia pre konkrétne témy, jemu prispôsobené odporúčania budú zo širších tém. Platí, že používateľ dostáva odporúčané iba najaktuálnejšie články z oblasti, ktorú určí jeho aktivita. Problém nadmernej špecializácie pri obsahovo zameraných odporúčacích systémom sme vyriešili práve týmto spôsobom. Desať odporúčaných článkov je kombinácia najnovších článkov v desiatich najrelevantnejších záujmoch. Takéto odporúčania pokrývajú široké spektrum, ktoré ovplyvňuje iba história používateľa. Ak je používateľ zaťažený na konkrétnu tému, napríklad hokej, potom odporúčania budú generované iba z tejto oblasti. Ak prejaví záujem o ďalšie témy, jeho oblasť sa v ostatných, menej relevantných stereotypoch rozšíri.

V prípade, že používateľ nemá žiadnu históriu, nie je možné odhaliť jeho záujem. Strom však pracuje so stereotypom záujmu, ktorý špecifikuje aj toto správanie a teda koreň stromu. Najrelevantnejší stereotyp tak vyhľadá vetvu s najnovším článkom a odporučí okolie tohto článku v strome. Toto riešenie nie je dostatočné, ale rieši inicializáciu odporúčaním najnovšej správy a správ k nej podobným. Proces hľadania záujmov v strome článkov a nasledovné odporúčanie môžeme zhrnúť v niekoľkých krokoch:

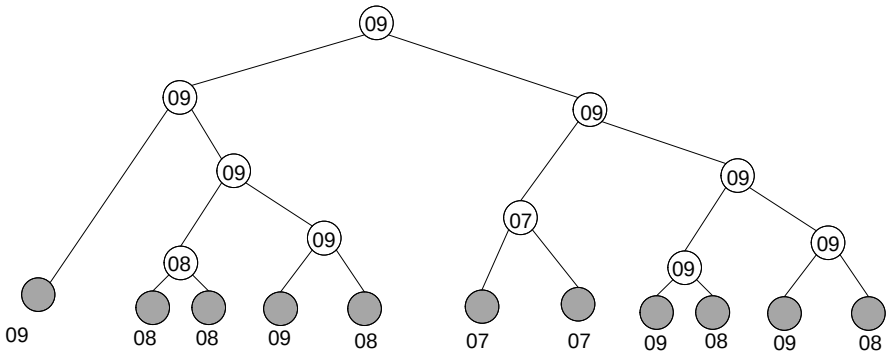
1. Výber článkov zobrazených používateľom.
2. Vytvorenie ciest ku koreňu stromu od každého článku (vytvorí sa nový strom).
3. Výpočet pomeru prečítaných, ku neprečítaným pre každú podvetvu nového stromu.
4. Zoradenie podľa relevantnosti určenej pomermi.
5. Postupný výber najaktuálnejších článkov z jednotlivých vetiev.
6. Vytvorenie mixu, obsahujúceho najnovšie, neprečítané články zo zoradených vetiev.

Pridaním možnosti hlasovať o záujme o článok, sa nám otvárajú ďalšie možnosti. V prípade, že čitateľ dostáva odporúčané články z oblasti, ktorá ho nezaujíma, môže vyjadriť svoj nezáujem a tieto články sa už ďalej nevyhľadávajú v procese objavovania záujmov používateľa a teda stereotypy záujmu zmenia poradie podľa relevancie. Hlasovanie, alebo podávanie spätnej väzby je však činnosť, ktorú používatelia vykonávajú v malej miere – je skôr obťažujúca. Preto by nemala byť kľúčová v procese odporúčania a preto plní skôr podpornú úlohu našej metódy pre tvorbu odporúčaní.

Čím viac čitateľ navštevuje stránku a číta články, tým viac je možné špecifikovať záujmy. Ak by používateľ využíval iba tento odporúčací systém, z logiky riešenia vyplýva, že by sa neustále zužovala jeho oblasť záujmov. Preto je vhodné, aby bolo toto odporúčanie kombinované s ďalšími prístupmi, alebo aby sa odporúčali aj náhodné články, ktoré priestor záujmov rozširujú.

6.3 UPREDNOSTNENIE AKTUÁLNYCH ČLÁNKOV

Jedným využitím stromovej reprezentácie je aj možnosť zakomponovať časovú informáciu. Čas pritom nemusí vystupovať ako atribút použitý pri tvorbe stromu. Proces rozhodovania pri ceste k najpodobnejšiemu vrcholu tak môže byť ovplyvnený časom. Vzhľadom na usporiadanie článkov podľa podobnosti, možno považovať každú množinu pod metadokumentom za skupinu podávajúcu rovnaký pohľad na obsahovú stránku. Každej tejto skupine preto priraďujeme aj ďalší atribút, určujúci aktuálnosť. Každá skupina je v tejto hierarchickej reprezentácii daná metadokumentom. Ak chceme rozšíriť informáciu o existencii aktuálneho článku, je nutné, aby sa stromom rozšírili časové údaje (obr. 17).



Obrázok 17. Ohodnotenie článku a vrcholov stromu časovou informáciou. Jednotlivé čísla pre ukážku predstavujú roky (čas sa však zvažuje až na minúty presne). Smerom vyššie v strome sa šíria aktuálnejšie časové známky. Takto sa zaručuje nezanedbanie jediného aktuálneho článku v podmnožine, ale prípadne vynechanie príliš starej oblasti.

Rozšírenie znamená určenie časovej známky každému metadokumentu, podľa novšej časovej známky z páru jeho potomkov. Pre každý metadokument tak vieme určiť čas pridania najaktuálnejšieho článku v skupine, ktorú určuje. Algoritmicky sa časová známka šíri rovnako ako ostatné vlastnosti dokumentu, platí teda nasledovné:

$Pre\ \forall v, v\ \in V, kde\ v\ je\ vrchol,\ V\ je\ množina\ vrcholov\ stromu,$
$pot(v) je\ množina\ všetkých\ potomkov\ vrcholu$
$a\ t(v) je\ časová\ známka\ vrcholu\ v\ platí, že$
$t(v) = \max_{1 \le n \le |pot(v)|} t(pot(v)_n)$

Počas rozhodovania o určení najpodobnejších článkoch sa potom zvažuje aj táto informácia, ktorá pomáha pri vynechaní určitej vetvy. V takom prípade používateľ, so záujmom o najaktuálnejšie dianie, nedostane množinu najpodobnejších článkov, ale množinu takých článkov, ktoré sú najpodobnejšie vzhľadom na zadané časové kritérium. Predpokladom je, že sa podobné články nachádzajú na jednej vetve v strome. V prípade odporúčaní vieme v rámci tejto vetvy jednoducho vyhľadať najnovšiu skupinu článkov a teda novinky v tomto stereotype záujmu. Ak predpoklad platí, potom môžeme jednoducho ignorovať staré články pri rozhodovaní. Ak tento predpoklad neplatí, potom sa v tejto množine nachádza článok novší, ktorý však svoju informáciu o aktuálnosti prešíril na svojich rodičov. V tomto prípade sa množina neignoruje a článok sa nestratí. Rozhodovanie možno opísať takto:

```
def vyberBlizsiehoPotomka(metadok, clanok, casobmedzenie)
      if metadok.potomok1.cas > casobmedzenie
            podobnost = vypPodobnost(metadok.potomok1, clanok)
            result = metadok.potomok1
      end
      if metadok.potomok2.cas > casobmedzenie
            if podobnost < vypPodobnost(metadok.potomok2, clanok)
                  result = metadok.potomok2
            end
      end
      return result
end
```

Samotný strom má schopnosť neobmedzene prijímať dokumenty. To znamená, že v strome sa časom vyskytnú aj také články, ktoré sú už zastarané. Najjednoduchšie riešenie je odstraňovanie článkov, ktoré sú

- príliš staré, vo vetve so starou časovou známkou,
- ich neaktuálnosť je potvrdená nezáujmom používateľov.

Pracujeme so stromom podobností. Každý vrchol má tiež spomínanú informáciu o najaktuálnejšom potomkovi. Najaktuálnejší potomok sa potom dá veľmi jednoducho objaviť preskúmaním stromu do hĺbky. Rovnako sa podobne dajú odhaliť aj vetvy, na ktorých už dlhšiu dobu nepribudli nové články. Celé vetvy potom vieme odstrániť, v takomto procese omladzovania stromu. Vetva je reprezentovaná jedným vrcholom, ktorý vystupuje ako článok a preto je možné ho zmazať aj s celou vetvou podobne ako na obrázku (obr. 14) v časti 5.3.

6.4 DISKUSIA

Hierarchia, ktorá vzniká využitím vzťahov podobností je binárnym stromom. Binárne vetvenie sme zvolili, aby sa jednotlivé operácie so stromom zjednodušili a tým zefektívnili. Rozhodovania sa pri hľadaní najpodobnejších článkov, alebo pri pridávaní nových článkov redukujú na dve možnosti, z ktorých sa vyberá vhodnejšia cesta postupne až ku listu stromu, ktorým je reálny článok.

Strom ako taký je vytvorený z usporiadaných článkov. Tieto články sú hierarchicky pospájané metadokumentmi, ktoré reprezentujú skupiny podobných článkov. Hrajú úlohu zhlukov a teda strom je vlastne hierarchia týchto zhlukov. Takéto skupiny podobných článkov potom môžu byť zo stromu jednoducho s nízkou zložitosťou získané. Vieme vytvárať zoznamy podobných článkov pre vyšetrovaný článok, ktoré sa podľa potreby môžu zväčšovať alebo zmenšovať – záleží na hĺbke stromu, ktorú zvolíme.

Keďže vetvy, či už väčšie alebo menšie obsahujú články, ktoré sú vzájomné podobné, tak strom od svojho koreňu až po jednotlivé listy rozdeľuje informačný priestor článkov. Čitateľov zaujímajú niektoré konkrétne témy, prípadne sa niektorým témam vyhýbajú. Na niektorej vetve sa teda nachádzajú informácie napríklad o hokeji, nakoľko tieto články majú obsahovo niečo podobné. Čitateľ potom bežným čítaním správ o hokeji prejavuje záujem o jednu takúto vetvu. Spolu s jeho inými záujmami vieme lokalizovať vetvy, ktoré určujú oblasti, ktoré ho zaujímajú. Spomínaný záujem o hokej sa potom prejaví ako vetva určená metadokumentom. Tento metadokument zastrešuje niekoľko článkov, ktoré čitateľ prečítal a niekoľko, ktoré neprečítal. Pomer týchto článkov môžeme chápať ako relevantnosť vetvy pre daného používateľa. Vetvy určené metadokumentom potom vieme zoraďovať vzhľadom na čitateľa a následne odporúčať najaktuálnejšie články, o ktoré má záujem.

Reprezentácia sa správa ako hierarchia zhlukov, čo znamená že jednotlivé vetvy obsahujú skupiny článkov vzájomne podobných. To však môže prinášať nedostatky a teda obmedzenia, ktoré vznikajú pretože neplatí tranzitívnosť relácie podobnosti. Môže sa tak stať, že k niektorému článku zo skupiny nenájdeme úplný zoznam jemu najpodobnejších článkov.

Keď hovoríme o tvorbe odporúčaní, je našim obmedzením predpoklad, že čitateľ zobrazuje väčšinu článkov, ktoré ho zaujímajú. Čím viac takýto čitateľ zobrazuje články, tým lepšie vieme lokalizovať záujmy, ako väčšinu článkov, ktoré zobrazil. Ak uvažujeme priemerného čitateľa, jeho aktivita nie je dostatočná na určenie správnych záujmov.

Napokon aktuálnosť článkov a ich skupín vieme reprezentovať, ale odoberanie vetiev v procese omladzovania stromu nie je dostatočné, ak by sme orezávali len najpočetnejšie vetvy. Algoritmom vieme jednoducho lokalizovať najaktuálnejší článok, ktorý sa na tejto vetve nachádza, ale starnúce, neaktuálne vetvy musíme odstraňovať od najpočetnejších až po jednotlivé články.

7 REALIZÁCIA METÓD

Na portál sme.sk, ktorý je zároveň prostredím realizácie našich metód pre tvorbu hierarchie článkov a tvorbu odporúčaní, pribudne počas dňa okolo 300 nových článkov, čo znamená, že bežný čitateľ nesleduje dianie v plnej miere. Čitateľ napokon nemá záujem o všetky informácie, avšak na ich triedenie rovnako nemá čas. Jeho ďalším problémom je okrem nových článkov aj starnutie článkov. Tým, že niektoré články už nie sú aktuálne, slabne o ne záujem, a mali by byť potlačené v prípadoch ak ich už tento používateľ čítal.

Tieto internetové noviny sú jedny s najväčších novín na Slovensku, pokrývajúce priestor spravodajstva a taktiež záujmov svojich čitateľov. Počet článkov robí prácu manuálnych zadávaní tém, alebo hľadaní podobností článkov nezvládnuteľnú. Pre tieto noviny je vhodná aplikácia automatizovaného hľadania podobností, prípadne jeho ďalšieho využitia napríklad pri tvorbe odporúčaní článkov čitateľom.

V tejto časti uvádzame hlavne prehľad prostredia, v ktorom sú jednotlivé časti implementované a podľa návrhu ponúkame postupne spresnenia, ktoré s realizáciou súvisia. Pre realizáciu sme zvolili platformu RubyOnRails s databázou MySQL, ktorá zastrešuje serverovú časť a teda jednotlivé algoritmy ako súčasti nami navrhnutých metód. Grafickým rozhraním a prezentáciou výsledkov sa nezaoberáme, nakoľko to nie je kľúčová záležitosť, avšak pre vytvorenie predstavy predvádzame náhľady rozhraní, ktoré používame. Grafické rozhranie je vytvorené kombináciou HTML a funkcií v jazyku javascript.

Články. Na sme.sk pribúda počas celého dňa množstvo článkov. Špeciálnym prípadom sú články, ktoré v návaloch pribúdajú v noci. Je to spôsobené paralelným udržovaním papierových novín, a teda podržania elektronického článku nezverejneného, až do doby jeho vytlačenia do dennej tlače. Sme.sk má mnoho autorov, ktorý píšu o rôznych témach, rôznym štýlom, v rôznom rozsahu. Články majú napriek rôznej zložitosti textu, alebo žánru svoju formu. Každý článok je pridaný s titulkom, krátkym opisom, informáciou o autorovi, časom zverejnenia, kľúčovými slovami, sekciou a kategóriou. Tieto informácie vznikli ručným zadávaním. Texty článku prebiehajú jazykovou korektúrou, založenou na kolaborácií čitateľov a oznamovaní chýb, preto sa pri popularite tohto denníka dá predpokladať spisovnosť textu a teda jeho správnosť.

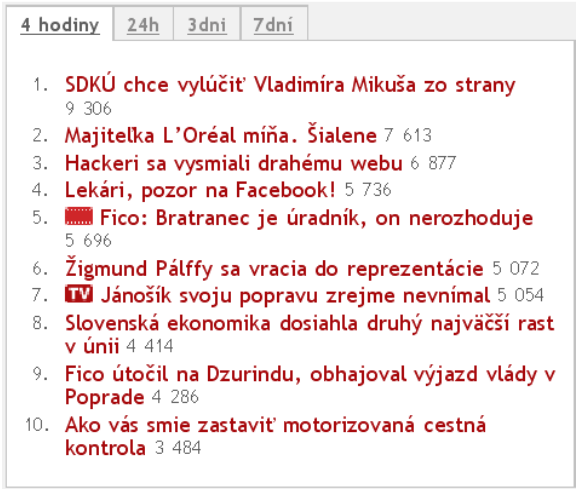
Články sú pridávané aj do RSS kanálov, takže je možné sledovať čerstvo pridané články. Okrem html kódu, z ktorého možno extrahovať informácie, sú dostupné metadáta (informácie o článku ako meno autora a pod.) aj prostredníctvom tohto kanála. Sme.sk taktiež poskytuje zjednodušený formát stránky, vhodnej pre automatizované sťahovanie. Táto verzia neobsahuje okolité prvky grafického rozhrania, čím sa zrýchľuje prístup k metadátam.

Používatelia. Tento portál navštevujú ľudia v práci, školách i domácnosti. Sú to ľudia mladí, starší s rôznorodými názormi a záujmami. Na stránku nie je nutné sa prihlásiť, čo zvyšuje dostupnosť. Problém nastáva v odlíšení rovnakých používateľov. Bez vyžiadania prihlasovania sa je čitateľ odlíšiteľný podľa cookie v prehliadači (informácia o používateľovi portálu), ktorú mu server priradí. Jedinečný identifikátor potom slúži na odlíšenie aktivít, ktoré používateľ vykonáva.

Ak však používateľ využíva portál v práci aj doma, nie je možné odlíšiť jeho osobu. Problém zrejme nie je kritický, pretože jeho osobnosť má viac rozmerov, závislých od prostredia a času, v ktorom sa nachádza. Čítanie novín doma je takto odlíšené od čítania v práci. Záznamy o činnostiach, ktoré portál uchováva sú viazané na osobnosť čitateľa. V tomto prípade nie je potrebné spájať monitorovanie používateľa doma a v práci, pretože môžeme predpokladať zmeny v záujmoch jednej osobnosti a teda odporúčania sa vzťahujú na konkrétne miesto a čas. Problémom už môže byť iba nízka aktivita používateľa napríklad v práci.

Záznamy o aktivitách používateľa obsahujú najmä informáciu o odkaze, ktorý si na sme.sk čitateľ zobrazil. Ďalej je zachytená informácia o čase, v ktorom používateľ článok zobrazil. Pomocou týchto záznamov môžeme sledovať záujmy jednej osobnosti a jeho aktivitách na webovej stránke. Nie je spoľahlivo možné určiť ako bol spokojný s obsahom, ktorý si zobrazil. Vieme však aspoň priblížiť jeho záujem.

Odporúčania. Využitím štatistík o počte návštev jednotlivých článkov v rôznych časových úsekoch sa na sme.sk v súčasnosti tvoria nepersonalizované odporúčania založené predovšetkým na sociálnych aspektoch [9]. Predpokladom je vhodnosť článku pre všetkých vzhľadom na väčšinový záujem. Takto však môžu významné udalosti prekryť informácie, ktoré sú pre jednotlivých čitateľov možno zaujímavejšie. Používateľ vidí záložky, ktoré obsahujú najčítanejšie články (obr. 18).



Obrázok 18. Záložky informujúce o popularite článkov medzi čitateľmi. Články sú zoradene podľa počtu prečítaní. Tiež je možné zobraziť štatistiku vo viac možných časových oknách.

Ďalším spôsobom odporúčania je zoznam súvisiacich článkov pod textom. Toto odporúčanie predpokladá záujem o podobné články a teda čitateľ, ak má záujem, môže prejsť na zobrazené odkazy. Problémom tohto riešenia v súčasnosti je, že sa manuálne musia zadať články, ktoré sú mu podobné.

7.1 SPRACOVANIE TEXTU

Navrhli sme metódu tvorby hierarchie článkov, pomocou ktorej vytvárame strom podľa podobnosti článkov, ktoré sú priebežne zadávané. Tieto články sa pridávajú v poradí v akom pribúdajú v čase. Pridanie článku znamená jeho stiahnutie a spracovanie do formy, ktorú sme navrhli na reprezentáciu dokumentu. Je možné zadať ľubovoľný dokument, povinný údaj je však text, podľa ktorého sa vykonáva samotné pridanie do stromu. Text je predmetom ďalšieho spracovania. Toto spracovania sa robí vždy pri pridaní nového článku nakoľko toto riešenie pracuje on-line. Vďaka tejto činnosti je možné pridať článok napríklad priamo z čítačky RSS, alebo aj po dávkach v časových intervaloch.

V prípade zadania html kódu zo stiahnutého článku, je nutné predspracovať tento text. Extrakciou metadát o článku, ako je čas pridania, titul a hlavne text článku je možné prejsť k ďalšiemu kroku práce s textom. Metóda nevyžaduje všetky metadáta článku ako sekcia, kategória, autor a podobne. Zaujímavé pre metódu porovnania podľa obsahu je iba titul a text. Atribút času extrahujeme pre udržanie informácie o aktuálnosti tohto článku.

Text samotný je v ďalšom kroku nutné rozdeliť na slová, v čom opäť aplikujeme regulárny výraz. Takto získané slová reprezentujú článok a môžu byť predmetom ďalšej analýzy [17]. Slová podľa zvolenej metódy redukcie zložitosti ďalej spracujeme algoritmami

- lematizér (základné formy)
- stemmer (korene slov)

Oba spôsoby redukujú zložitosť slovných foriem v slovenčine, normalizujú text (malé písmená, stop slová). Pre lématizer sme použili slovník, obsahujúci preklad slovnej formy na lému. Tento slovník premieta približne 500 tisíc foriem, na množinu 50 tisíc základných foriem slovenského jazyka. Toto spracovanie významne redukuje zložitosť vektora opisujúceho článok. V skutočnosti však 175 tisíc článkov stiahnutých zo sme.sk podľa návštev z novembra 2009, obsahuje takmer 900 tisíc jedinečných slov. Rozdiel oproti kapacite slovníka spôsobujú mená, názvy, skratky, ale aj preklepy, citoslovce a nespisovné výrazy. Hovoríme o redukcii

- pozitívnej (odstránenie nežiadúcich slov)
- negatívnej (strata slov významných pre obsah článku)

Práve pre redukciu rozmeru a bezstratovosť významných slov bol vytvorení štatistický stemmer [20]. Tento spôsob vygeneruje vlastný slovník prekladu formy na koreň slova pomocou jedinečných 900 tisíc slov, ktoré sa objavili v článkoch z domény spravodajstva. Stemmer pracuje postupne ako zlučovač najpodobnejších slov na ich spoločný základ. Podobnosť určujeme princípom Hammingovej vzdialenosti. Výpočet vzdialenosti sme však museli upraviť podľa pravidiel gramatiky slovenského jazyka. To znamená, že zmena na konci slova je viac očakávaná, a preto aj zmeny koncovky majú na podobnosť menší vplyv.

Spracovanie článkov sme realizovali v tejto postupnosti:

1. Vytvorenie bázy slov (slovník vlastností, mapovač slov na vlastnosti).
2. Stiahnutie článkov (vytvorenie stahovača aktuálne pridaných článkov).
3. Predspracovanie (extrahovanie slov, času, titulu a ďalších potrebných metadát).
4. Analyzovanie vlastností (normovanie slov, filtrovanie, tvorba vektora vlastností).

Články sme spracovali v dávke, aby sme získali slovník a vedeli tak určiť dôležitosť slov, ktoré ďalej využívame pri pribežnom sťahovaní. Po stiahnutí článkov sme boli schopný vygenerovať slovník, ktorý obsahuje v prvom rade slová z tvaroslovníka pre slovenský jazyk a následne slová, ktoré v ňom neboli nájdené, ale boli vytvorené našim štatistickým stemmerom. Zo slovníka sme ďalej vylúčili slová, ktoré sa často opakujú a preto nemajú obsahový význam pre články textov. Takto vytvorený slovník obsahuje mená, značky, ale aj spisovné slová z jazyka slovenského. Celý slovník je potom potrebný pri spracovaní jednotlivých článkov a pomáha pri vytvorení množiny slov ohodnocujúcich článok.

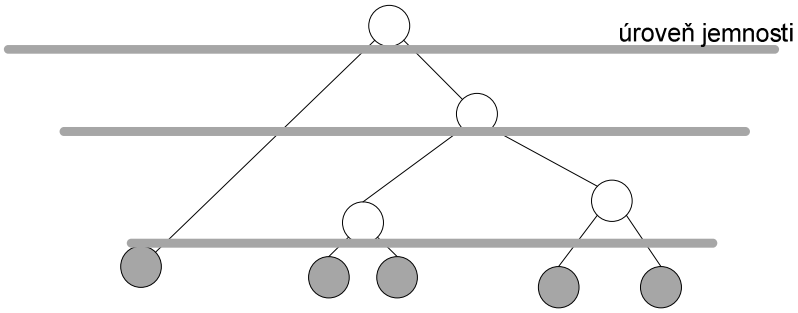
7.2 TVORBA STROMOVEJ ŠTRUKTÚRY

Vytvorili sme softvérový nástroj, ktorý plno zvláda funkcionalitu napĺňania stromu. Strom je uložený v pamäti programu, pričom nepotrebuje prístup k databáze. Strom ako inštancia dokáže pracovať s rôznymi stratégiami výpočtu podobnosti. V prototype boli implementované dva spôsoby zisťovania podobností:

- množina slov a percentuálne vyjadrenie počtu spoločných,
- frekvencia slov a kosínusová podobnosť.

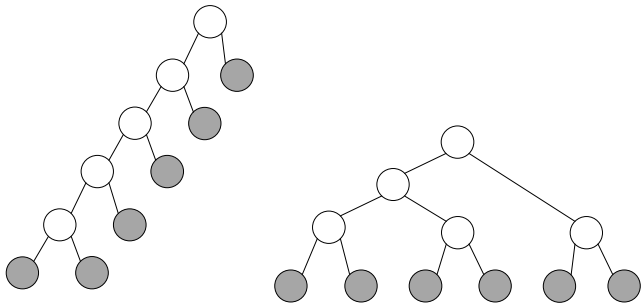
Ak využívame kosínusovú podobnosť, pracujeme s vektormi jednej dĺžky. Táto dĺžka je určená počtom slov článkov v báze slov získanej z dostatočne veľkej množiny článkov. Dĺžka potom priamo ovplyvňuje rýchlosť porovnania. Napriek optimalizácii kosínusovej podobnosti pre vyberanie bližšieho z dvojice potomkov je fakt, že vektory sú riedke (obsahujú veľa núl), a preto je tento čas dlhší ako pri porovnaní prvkov iba podľa množiny spoločných slov. Podobnosť podľa počtu rovnakých slov je rýchlejšou a bola vybraná hlavne pre potreby testovania na väčších počtoch článkov. Podobnosť samotná nie je predmetom skúmania. Snažíme sa najmä reprezentovať takto vzniknuté vzťahy, aby boli vhodne usporiadané pre ďalšie hľadanie v množine článkov.

Tvorba stromu vychádza najmä zo spôsobu porovnania, a je otvorená pre iné metódy porovnania. Pri zmene stratégie porovnania treba implementovať vytvorenie vektora opisujúceho článok, výpočet rodiča a metódu samotného porovnania. Strom potom túto funkcionalitu ďalej využíva pre hľadanie podobných článkov. Podobne ako pri tvorbe zhlukov vieme poskytnúť skupiny podobných článkov. Vytváraním rezov vieme deliť strom na vetvy, ktoré potom obsahujú články podobné na danej úrovni jemnosti (obr. 19).



Obrázok 19. Skupiny podobných článkov. Čiary pod jednotlivými metadokumentmi vizuálne rozdeľujú strom na vetvy, ktoré obsahujú podobné články. Jemnosť pritom môže byť určená na rôznych úrovniach.

Ďalším pozorovaním sme zistili malú vzájomnú podobnosť článkov na úrovni textu. To znamená, že pri porovnaní vychádzajúceho z obsahu článku nastáva problém "nulovej podobnosti". Prakticky to znamená, že vektory vlastností sú na seba väčšinou kolmé v priestorovom ponímaní. Pri stavbe stromu sa tento jav ukáže ako veľmi hlboký strom. Preto bola implementovaná rozširujúca vlastnosť a to vyvažovanie stromu. Princíp spočíva v pridaní nového prvku nie rozvetvením hrany na mieste zistenia, ale pridaním prvku pod zistené miesto tak, aby strom bol čo najširší (obr. 20).



Obrázok 20. Porovnanie dvoch metód tvorby stromu. Do stromu sú pridávané články spolu nesúvisiace. Hlboký strom (vľavo) je korektný, široký strom (vpravo) je rýchly. Široký strom však môže obsahovať články na rovnakej úrovni (priamy potomkovia jedného vrcholu), avšak spolu vôbec nesúvisia. Široký strom ďalej očakáva prehlbovanie jednotlivých vetiev, ak sa objavia podobné články. (hlboký je N porovnaní, široký je $\log_2 N$ porovnaní)

7.3 HĽADANIE STEREOTYPOV ZÁUJMU POUŽÍVATEĽA

Vytvorili sme softvérový nástroj, prostredníctvom ktorého sme schopní odhaľovať jednotlivé stereotypy záujmu a ich hierarchiu. Tieto záujmy vie taktiež zoradiť podľa relevancie a vybrať najaktuálnejšie články, ktoré v týchto stereotypoch vznikajú. Algoritmus využíva strom podobností. Najprv zoberie posledných sto článkov, ktoré používateľ prečítal a vyhľadá ich v strome. Tieto listy potom najkratšou cestou spojí s koreňom stromu. Vznikne tak, akoby ďalší strom, ktorý obsahuje iba prečítané články a vrcholy, ktoré ich spájajú podľa podobnosti. Tieto vrcholy potom označíme ako stereotypy záujmu. Vieme ich tiež zoradiť podľa pomeru článkov, ktoré v strome podobností prečítal

a ktoré ešte nie. Takto vznikne zoznam stereotypov, z ktorých vyberieme vždy najaktuálnejšie články a odporúčame.

Zo zoradených stereotypov vyberáme vždy iba prvý najaktuálnejší článok. Takto sa snažíme pokryť širšie spektrum záujmov, ktoré používateľ má. Články a záznamy o čítaní sa pritom pridávajú pribežne, prípadne v krátkych intervaloch. To znamená, že odporúčania preto v tomto prototype vymenia až po pridaní nových záznamov.

7.4 VÝSTUP METÓD PRE REPREZENTÁCIU VZŤAHOV A TVORBU ODPORÚČANÍ

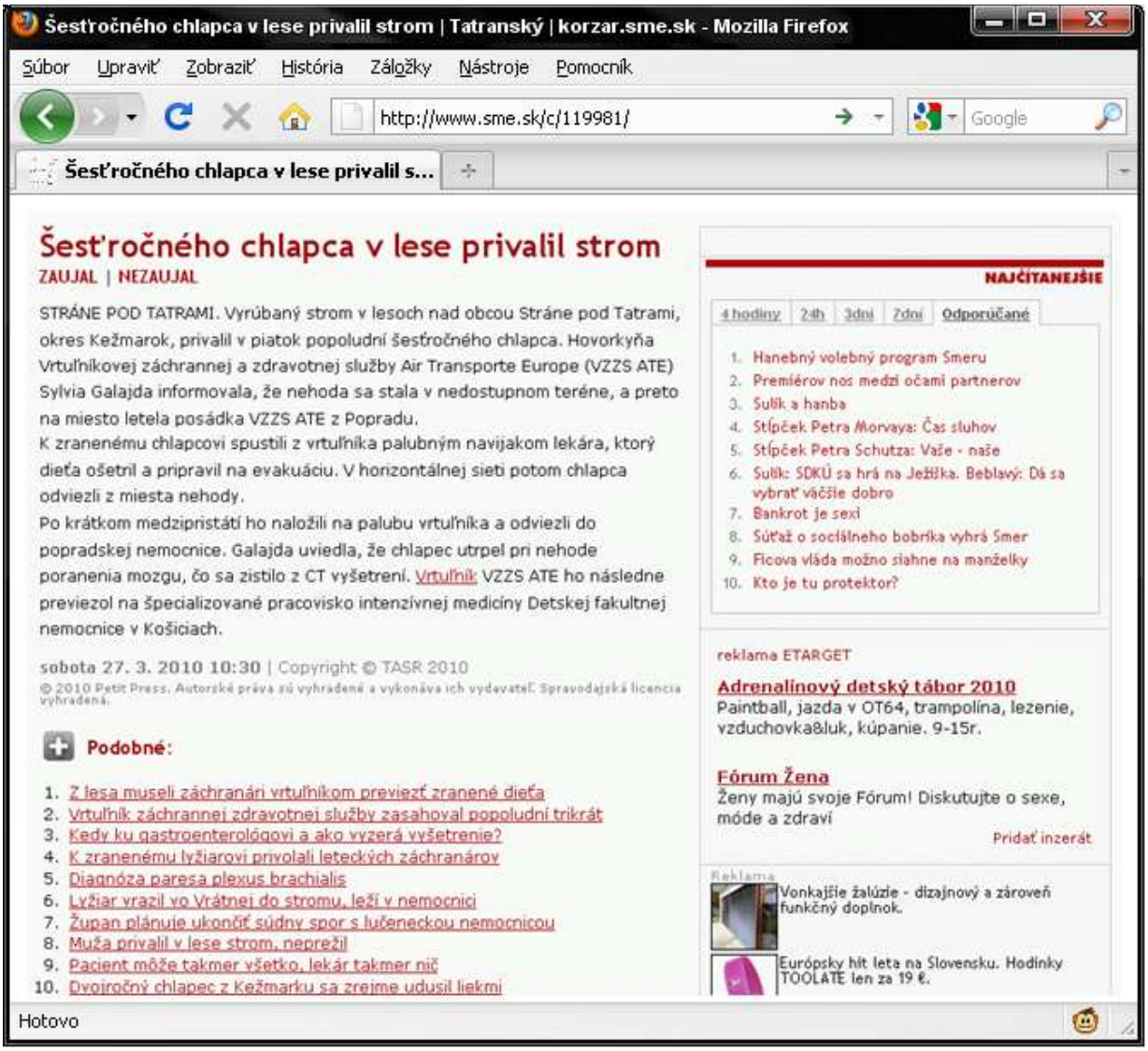
Reprezentácia vzťahov podobností a metódy jej tvorby a modifikácie už spĺňajú funkcionalitu, ktorá zabezpečuje výstup v podobe množiny výsledkov najbližších prvkov k prvku zadaného do stromu a odporúčania článkov podľa histórie čitateľa. V prípade potreby vyhľadania množiny podobných článkov je vstupom identifikátor článku. V prípade odporúčania je to identifikátor čitateľa. Výstupom predchádza aj priebežné spracovanie článkov, ktoré sú postupne zverejňované.

Prostredníctvom riešenia pre reprezentáciu vzťahov medzi článkami postupne sťahujeme najnovšie články dostupné na sme.sk. Tieto články potom spracujeme slovníkom, aby sme získali množinu slov. Táto množina je potom použitá pri rozhodovacom procese hľadania miesta v strome. Postupne sa takto vytvára strom podobných článkov.

Pri vyhľadanie podobných článkov pracujeme už s existujúcim stromom. Podľa zadaného identifikátora môžeme nájsť článok už uložený v strome. Od vyhľadaného článku potom môžeme postupovať po rodičoch vyššie až kým nezískame väčšiu množinu článkov, ktoré spája. V tejto množine je samozrejme aj vyšetrovaný článok, ale aj jeho okolie. Túto množinu ďalej zoradíme, aby sme získali N najbližších článkov podľa poradia. Samotný strom teda zabezpečuje formu heuristiky, ktorá filtruje veľký počet dokumentov a na výstup dáva rádovo omnoho menšiu množinu podobných článkov. Táto množina sa potom zoradí rýchlejšie ako celá báza článkov v strome a zobrazí sa pre vyšetrovaný článok (obr. 21).

Odporúčanie článkov podobne ako pri vyhľadaní podobných článkov. Proces opäť inicializuje používateľ, avšak s dotazom sa odosiela identifikátor používateľa. V databáze sa potom vyhľadajú záznamy histórie tohto používateľa, pričom sa články videné používateľom mapujú na články v strome podobných článkov. Získame odporúčanie a tie sa potom presunú na obrazovku používateľa.

Integráciu samotnú sme riešili aj v prípade stránky sme.sk obohatením grafického rozhrania, ktoré využívajú. V podobne rozšírení pre rôzne prehliadače tak môže požívateľ okrem prehliadania podobných článkov prezerať aj články, ktoré sú mu odporúčané podľa jeho histórie. V oboch prípadoch sme pridali desať článkov, aby sme nezahltili používateľa a pridali sme možnosť vyjadrenia záujmu (pod nadpisom), ktorá pomôže pri overovaní riešenia, ale aj ďalších našich predpokladov.



Obrázok 21. Pohľad na zobrazenie článku na sme.sk s obohatením o odporúčania (vpravo hore v pôvodnej časti NAJČÍTANEJŠIE) a o podobné články (vľavo dole, namiesto pôvodných ručne zadávaných súvisiacich článkov).

8 VYHODNOTENIE METÓD

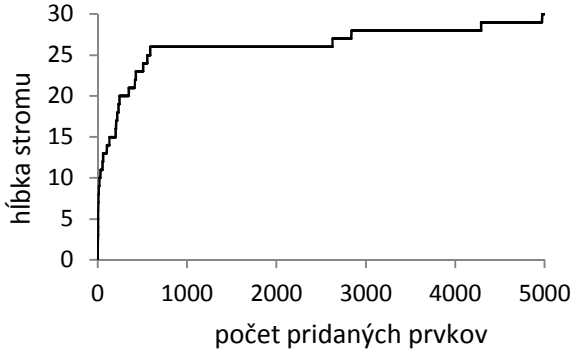
Overenie vlastností metód pre prácu s reprezentáciou a tvorbu odporúčaní môžeme uskutočniť viacerými spôsobmi. Overujeme časť pre odporúčanie článkov a časť pre hľadanie podobných článkov. Ako celok teda hodnotíme naše riešenie pre:

- efektívnosť reprezentácie podobných článkov
- určenie podobných článkov
- generovanie odporúčaní podľa modelu používateľa

8.1 EFEKTÍVNA REPREZENTÁCIA

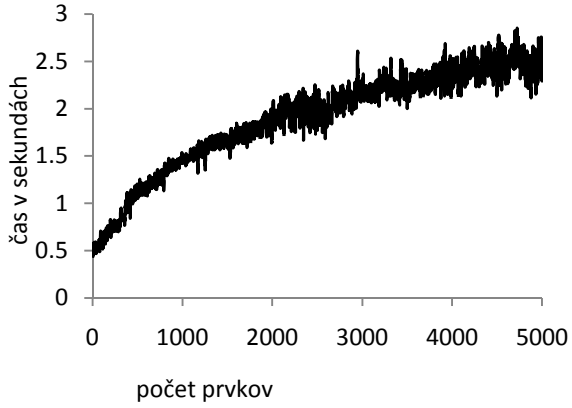
Overenie efektívnosti riešenia znamená v prvom rade overenie zložitosti tejto reprezentácie. Našou hypotézou je, že toto riešenie má prijateľnejšiu zložitosť ako maticové, alebo množinové riešenia toho istého problému. Porovnanie efektívnosti výpočtu podobnosti jednotlivých dokumentov nie je predmetom experimentov, nakoľko tento výpočet je rovnaký a výsledky by taktiež boli rovnaké. Experimenty sme vykonali na rovnakej vzorke päťtisíc náhodných článkov.

Efektívnosť riešenia hodnotíme najmä z predpokladov, že stromová štruktúra poskytuje logaritmickú zložitosť. Uskutočnili sme preto experiment, ktorý odhaľuje, že hĺbka stromu narastá logaritmicky s počtom článkov. To potvrdzuje predpoklad, že s narastajúcim počtom článkov pridaných do stromu sa zložitosť pridávania nebude zvyšovať lineárne (obr. 22).



Obrázok 22. Logaritmická hĺbka stromu v závislosti od počtu pridaných prvkov.

Ďalším predpokladom je, že časy potrebné na pridávanie rastú logaritmicky a nie lineárne s počtom vložených článkov. Tento predpoklad overujeme ako zobrazenie závislosti počtu pridaných prvkov a času potrebného na vloženie nového prvku (obr. 23).



Obrázok 23. Logaritmický čas, potrebný pre pridanie nového prvku v závislosti od počtu prvkov reprezentovaných v strome.

8.2 PODOBNOSŤ ČLÁNKOV V SKUPINÁCH

Rozhodli sme sa vyhodnotiť úspešnosť metódy pre hľadanie podobných prvkoch vo vytvorenej reprezentácii. Hypotéza je, že najbližšie články v hierarchickej reprezentácii podobností sú zároveň najpodobnejšie a že referenčná metóda odhalí chybu, s ktorou najpodobnejšie články získavame. Vybrali sme referenčnú metódu, s ktorou môžeme výsledky porovnať. Tu je však zaujímavé, aké atribúty hodnotiť. Ak berieme v úvahu 100% úspešnosť metódy, ktorá porovnáva podobnosť prvkov ako každý s každým, potom v tomto poli hodnotenia experiment dopadne lepšie pre referenčnú metódu. Ak sa však porovnáva efektívnosť metódy, potom naše riešenie dosahuje lepšie výsledky.

Taktiež záleží aj na funkcii výpočtu podobnosti. Porovnávať metódy, využívajúce iné metriky zisťovania podobnosti s metódou, ktorá dokáže prijať aj tieto metriky nemá význam. Metriku porovnania v referenčnej metóde preto treba vybrať tak, aby korešpondovala s metódou hierarchickej reprezentácie podobnosti.

Postup tohto experimentu je taký, že spracujeme množinu článkov o počte päťtisíc našou a referenčnou metódou. Referenčná metóda pracuje ako hrubá sila, ktorá porovná každý prvok s každým, ak má záujem zobraziť podobné články. Porovnaní s touto metódou získavame veľmi dobrú predstavu o kvalite našej metódy, nakoľko porovnávame naše riešenie s veľmi komplexným a výpočtovo dlhým riešením (rádovo 100 krát dlhšie pre našu vzorku).

Pre obe metódy platí, že pre každý prvok vzorky vyhľadáme najpodobnejších N článkov. Zoberieme prvých desať z tejto množiny a porovnáme ich navzájom. Keďže referenčná metóda má 100% úspešnosť, odvíja sa od nej aj výpočet presnosti našej reprezentácie (tab. 2)

Najpodobnejších N	10	50	100	150	200	250	300	350	400	450	500
Presnosť %	27.3	42.8	59.6	65.2	78	81.8	85.4	89.6	90.6	91.6	97

Tabuľka 2. Porovnanie prvých presnosti s referenčnou metódou. Porovnaných je prvých desať prvkov oboch metód pričom obe riešenia vyberali z množiny päťtisíc článkov najpodobnejších N článkov postupne ku 100 náhodným článkom.

Naše riešenie je podľa experimentu schopné vyhľadať podobné články, avšak s výraznou chybou na prvých priečkach. Toto je spôsobené najmä tým, že naše riešenie hľadá skupiny podobných článkov. To je ovplyvnené faktom, že relácia podobnosti nie je tranzitívna. A teda neplatí, že ak A, B sú si podobné a B, C sú si podobné potom C, A sú si podobné. Takto sa v skupinách objavia aj prvky, ktoré nemajú vzájomnú podobnosť. Zväčšovaním skupiny však postupne pribúdajú najpodobnejšie články, čo v zásade dokazuje blízkosť podobných článkov.

8.3 TVORBA PERSONALIZOVANÝCH ODPORÚČANÍ

Generovanie odporúčaní vo forme článkov, o ktoré by mohol mať používateľ záujem závisí od preferencií čitateľa a od podobnosti článkov. Zaujímavý je opäť názor používateľa, ktorý tento výsledok využíva. Možno priamo sledovať využívanie odkazov, ktoré sa odporúčajú. Hypotézou je, že odporúčania pokrývajú záujmy používateľa. Pre overenie predpokladov sme vytvorili experiment, ktorý využíva históriu tisíc aktívnych čitateľov. Z histórie sa potom vyberie interval, ktorý sa rozdelí na dve časti. Z prvého obdobia sa pomocou našej metódy vygenerujú odporúčania a porovnajú sa s druhým obdobím histórie podobne ako to uviedli v námetoch na budúcu prácu Bogers aBosch [11].

Odporúčania však nemá význam hodnotiť podľa presných článkov, ktoré si používateľ prečítal v druhom období. Pretože takto by sme hodnotili predikčné metódy a nie odporúčací systém. Čitateľ teda za bežného čítania neprečíta iba tie články, ktoré ho zaujímajú. Ak aj prečíta, potom je možné, že v tejto téme sa nachádzajú i ďalšie, ku ktorým sa nedostane z pocitu oboznámenia sa s témou.

Porovnali sme preto kombinácie sekcie a kategórie článku. Sekcia i kategória je informácia, ktorá je získaná spolu s článkom. V použitej dátovej vzorke 15 tisíc článkov existuje 427 jedinečných kombinácií. Kombinácie teda môžu hovoriť o záujmoch používateľa, bez potreby porovnania článkov podľa ich presného identifikátora. Porovnanie malo pre rôzne časové okná rôznu úspešnosť (tab. 3).

Okno pre porovnanie	1 hod.	4 hod.	10 hod.	24 hod.	48 hod.
Presnosť	40	49	56	58	59
Pokrytie	71	60	44	32	25

Tabuľka 3. Presnosť a pokrytie pre meniaci sa testovací interval.

Z experimentu vyplýva, že v priebehu jednej hodiny čitateľ neprečítal mnoho článkov, a preto boli lepšie pokryté. Čím viac článkov čitatelia v čase prečítali, tým sa pokrytie znižovalo, nakoľko odporúčame iba desať článkov. Presnosť sa správa opačne, takže rastie s časom a väčšou možnosťou určiť správnu kombináciu sekcie a kategórie.

Ďalším experimentom, potvrdzujúcim úspešnosť odporúčaní je zobrazenie týchto výsledkov skutočným používateľom. Tí majú možnosť ohodnotiť odporúčanú položku ako zaujal a nezaujal. V rámci možností sme oslovili 10 čitateľov sme.sk, ktorí ohodnotili 88 odporúčaní. Z týchto odporúčaní bolo ako nevhodných označených 26, čo znamená približne 70% úspešnosť. Tento experiment je vhodné opakovať s väčším počtom čitateľov, aby tieto výsledky nadobudli štatisticky významnejšie čísla.

9 ZÁVER A BUDÚCA PRÁCA

Metódou hierarchickej reprezentácie vzťahov v dynamickom prostredí článkov riešime problémy, ktoré bežne nastávajú s charakterom týchto informácií. Tieto problémy nie sú jedinečné pre oblasť internetových novín, ale aj databáz odborných článkov a iných textových dokumentov pridávaných v čase. Mimo oblasti textových dokumentov môžeme prejsť až na podobnosť objektov ako je zvuk, alebo obraz. Rozšíriteľnosť do iných domén zabezpečuje aj prístup, ktorý môže použiť iné metódy získavania podobnosti párov objektov. Nahradením výpočtu podobnosti tak možno vkladať do hierarchie aj objekty reálneho sveta ak sa dajú opísať vektorom. My sme sa však sústredili na doménu internetových novín, kde ako entity vystupujú články.

Vytvorili sme softvérový nástroj, ktorý je schopný usporiadať články do hierarchie podľa vzájomných vzťahov podobnosti. Inšpirovali sme sa pri tom riešeniami z odbornej literatúry, hlavne algoritmami pre tvorbu hierarchických zhlukov. Nami vytvorené riešenie tak vytvára skupiny od najmenších dvojíc, až po jednu skupinu obsahujúcu všetky prvky. Efektivita bola hlavná úloha, ktorú sme si určili a preto vieme reprezentovanú podobnosť spätne získať z nami vytvorenej štruktúry. Vyhľadanie podobných článkov k článku, ktorý je v hierarchii umiestnený sa tak zjednodušuje, čo nám otvára ďalšie možnosti využitia tejto reprezentácie. Usporiadanie článkov vytvára skupiny podobných článkov, podobne ako pri zhlukovaní, a umožňuje tak na ľubovoľnej úrovni zobraziť články s podobným obsahom. Uzly stromu ďalej obsahujú aj časovú informáciu, ktorá pomáha pri rýchlom hľadaní najnovších článkov v rámci takýchto skupín.

Reprezentácia podobností sa ďalej využíva pre odporúčanie článkov čitateľovi. Články odporúčame podľa histórie, ktorá sa zaznamenáva pre každého používateľa. História obsahuje články, ktoré si čitateľ zobrazil. Tieto články potom mapujeme na skupiny určené v našej reprezentácii. Články teda na rôznej úrovni podobnosti určia vetvy stromu, ktoré sú pre používateľa zaujímavé. Výsledné vetvy potom vieme zoradiť podľa relevantnosti (pomer prečítaných ku neprečítaným) a generovať odporúčania. Odporúčania sa generujú z viacerých stereotypov záujmu, aby tak vytvorili mix, ktorý pokrýva preferencie čitateľa. Rozhodli sme sa vyberať iba najaktuálnejšie články z jednotlivých stereotypov, aby sme nezahlcovali čitateľa.

Naše riešenie je možné využívať ako rozšírenie rôznych prehliadačov, pričom používateľovi sa obohatí samotná stránka sme.sk, kde sa mu zobrazia odporúčané články pre jeho osobu a články podobné k tomu, ktorý práve číta. Používateľ môže odporúčaniam prideliť svoj hlas, ktorý určuje, či je pre neho článok zaujímavý alebo nie. Toto rozšírenie môže pomôcť k úprave záujmov, ale i k vyhodnoteniu predpokladov. Celé riešenie sme overovali postupne od hierarchie až po vyhodnotenie záujmu používateľa o článok. Využívame pritom skutočné články, a skutočné záznamy histórie. Tak sme mohli simulovať správanie používateľov a pozorovať úspešnosť našej metódy.

Naše riešenie je súčasťou širšieho projektu SME-FIIT, preto mnohé časti, ako sťahovanie článkov, alebo histórie používateľov nie sú súčasťou tejto práce. Naše riešenie preto tieto podporné mechanizmy využíva, čo nám umožnilo sústrediť sa na nami určené ciele. Ohraničeniami pritom ostávajú spracovanie iných prvkov ako sú textové dokumenty a taktiež výpočet samotnej podobnosti. Tieto časti neboli predmetom tejto práce, preto neboli ani predmetom našej analýzy. V našej práci

sme však využili riešenia, ktoré stručne opisujeme pričom sme umožnili jednoduchú výmenu týchto prevzatých častí.

Hlavným prínosom je návrh efektívnej reprezentácie množiny dokumentov, využitím vzťahov podobnosti. Vytvorenie hierarchie podobnosti článkov, je potom ďalej vďaka efektivite využiteľná pri tvorbe obsahových odporúčaní. Námetom na budúcu prácu je vytvorenie kombinácie obsahového a sociálneho odporúčacieho systému postaveného na týchto základoch. Používateľovi totiž vieme identifikovať záujmy, ako vetvy stromu. Tieto vetvy však môžu byť spoločné pre viac používateľov a teda napríklad v prípade výberu odporúčania z konkrétnej vetvy sa odporučia tie, k ktoré sú z nej najčítanejšie. Takto sa do obsahového odporúčania vnesie sociálny princíp.

Našou ďalšou snahou je preto okrem ďalšieho výskumu tejto oblasti aj nasadenie a sprístupnenie jednotlivých častí, aby tak čitatelia, alebo aj autori článkov využívali poskytované výhody.

LITERATÚRA

[1] Adomavicius, G. and Tuzhilin, A. 2005. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. IEEE Trans. on Knowl. and Data Eng. 17, 6 (Jun. 2005), 734-749.

[2] Aggarwal, C. C. 2001. On the effects of dimensionality reduction on high dimensional similarity search. In Proc. of the 20th ACM SIGMOD-SIGACT-SIGART Symp. on Principles of Database Systems (CA, USA). PODS '01. ACM, New York, NY, 256-266.

[3] Aggarwal, C. C., Hinneburg, A., and Keim, D. A. 2001. On the Surprising Behavior of Distance Metrics in High Dimensional Spaces. In Proc. of the 8th int. Conf. on Database theory (2001). J. V. Bussche and V. Vianu, Eds. Lecture Notes In Computer Science, vol. 1973. Springer-Verlag, London, 420-434.

[4] Aggarwal, C. C. and Yu, P. S. 2000. Finding generalized projected clusters in high dimensional spaces. In Proc. of the 2000 ACM SIGMOD int. Conf. on Management of Data (Texas, USA, 2000). SIGMOD '00. ACM, New York, NY, 70-81.

[5] Aggarwal, C. C. and Yu, P. S. 2001. On Effective Conceptual Indexing and Similarity Search in Text Data. In Proc. of the 2001 IEEE international Conference on Data Mining (2001). N. Cercone, T. Y. Lin, and X. Wu, Eds. ICDM. IEEE Comp. Society, Washington, DC, 3-10.

[6] Ahn, J., Brusilovsky, P., Grady, J., He, D., and Syn, S. Y. 2007. Open user profiles for adaptive news systems: help or harm?. In Proceedings of the 16th international Conference on World Wide Web (Banff, Alberta, Canada, May 08 - 12, 2007). WWW '07. ACM, New York, NY, 11-20.

[7] Arnaldo J. Abrantesy and Jorge S. Marquesz 1998. A Method for Dynamic Clustering of Data. In Proc. of the 9th British Conference, University of Southampton, UK, 1998, 154-163.

[8] Banerjee, A., Krumpelman, C., Ggosh, J., Basu, S., Mooney, R. J. 2005. Model-based overlapping clustering. In Proc. of the 11th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD). 532–537.

[9] Barla, M., Kompan, M., Suchal, J., Vojtek, P., Zeleník, D., Bieliková, M., 2010. News recommendation. In Proc. of the 9th Znalosti, Jindrichuv Hradec, ČR, 171-174.

[10] Billsus D., Pazzani, M. 2000. User Modeling for Adaptive News Access. User Modeling and User-Adapted Interaction, vol. 10, nos. 2-3, 147-180.

[11] Bogers T., Bosch A. 2007. Comparing and evaluating information retrieval algorithms for news recommendation. In Proceedings of the 2007 ACM conference on Recommender systems, Minneapolis MN, USA, October 2007, 141-144.

[12] Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., Sartin, M. Combining Content-Based and Collaborative Filters in an Online Newspaper. In Proc. ACM SIGIR '99 Workshop Recommender Systems: Algorithms and Evaluation, Aug. 1999.

[13] Das, A. S., Datar, M., Garg, A., and Rajaram, S. 2007. Google news personalization: scalable online collaborative filtering. In Proceedings of the 16th international Conference on World Wide Web (Banff, Alberta, Canada, May 08 - 12, 2007). WWW '07. ACM, New York, NY, 271-280.

[14] Deerwester, S., Dumais, S., Furnas, G., Landauer, T. and Harshman, R., Indexing by latent semantic analysis. Journal of the American Society for Information Science. 41. 391-407.

[15] Dhillon, I. S. and Modha, D. S. 2001. Concept Decompositions for Large Sparse Text Data Using Clustering. *Mach. Learn.* 42, 1-2 (Jan. 2001), 143-175.

[16] Gao, J. and Zhang, J. 2005. Clustered SVD strategies in latent semantic indexing. *Inf. Process. Manage.* 41, 5 (Sep. 2005), 1051-1063.

[17] Garabík R., Gianitsová L., Horák A., Šimková M. Tokenizácia, lematizácia a morfologická anotácia Slovenského národného korpusu. 2004. Interný materiál.

[18] Guinepain, S. and Gruenwald, L. 2005. Research issues in automatic database clustering. SIGMOD Rec. 34, 1 (2005), 33-38.

[19] Husbands, P., Simon, H., & Ding, C. 2001. On the use of singular value decomposition for text retrieval. In Computational Information Retrieval (SIAM), Philadelphia, PA. 45–156.

[20] Hull, D.A. Stemming Algorithms – A Case Study for Detailed Evaluation. Journal of the American Society for Information Science. 47, 70-84. 1996.

[21] Jain, AK and Murty, MN and Flynn, PJ (1999) Data Clustering: A Review. In: ACM Computing Surveys. 31, 264-323.

[22] Kohonen, T. 1990. The self-organizing map. In Proc. of the IEEE, (1990), 78, 1464-1480.

[23] Konstan, J. A., Miller, B. N., Maltz, D., Herlocker, J. L., Gordon, L. R., and Riedl, J. 1997. GroupLens: applying collaborative filtering to Usenet news. Commun. ACM (1997). 40, 77-87.

[24] Parsons, L., Haque, E., and Liu, H. 2004. Subspace clustering for high dimensional data: a review. SIGKDD Explor. Newsl. 6, 1 (2004), 90-105

[25] Plate, T. A. 2003. Holographic Reduced Representation: Distributed Representation for Cognitive Structures. CSLI Publications.

[26] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. 1994. GroupLens: an open architecture for collaborative filtering of netnews. In Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work (Chapel Hill, North Carolina, United States, October 22 - 26, 1994). CSCW '94. ACM, New York, NY, 175-186.

[27] Steiner, M. and Biersack, E. W. 2005. DDC: a dynamic and distributed clustering algorithm for networked virtual environments based on P2P networks. In Proc. of the 2005 ACM Conf. on Emerging Network Experiment and Technology (Toulouse, France, 2005). CoNEXT '05. ACM, New York, NY, 288-289.

[28] Şerban, G. and Câmpan, A. 2008. Hierarchical Adaptive Clustering. Informatica 19, 1 (2008), 101-112.

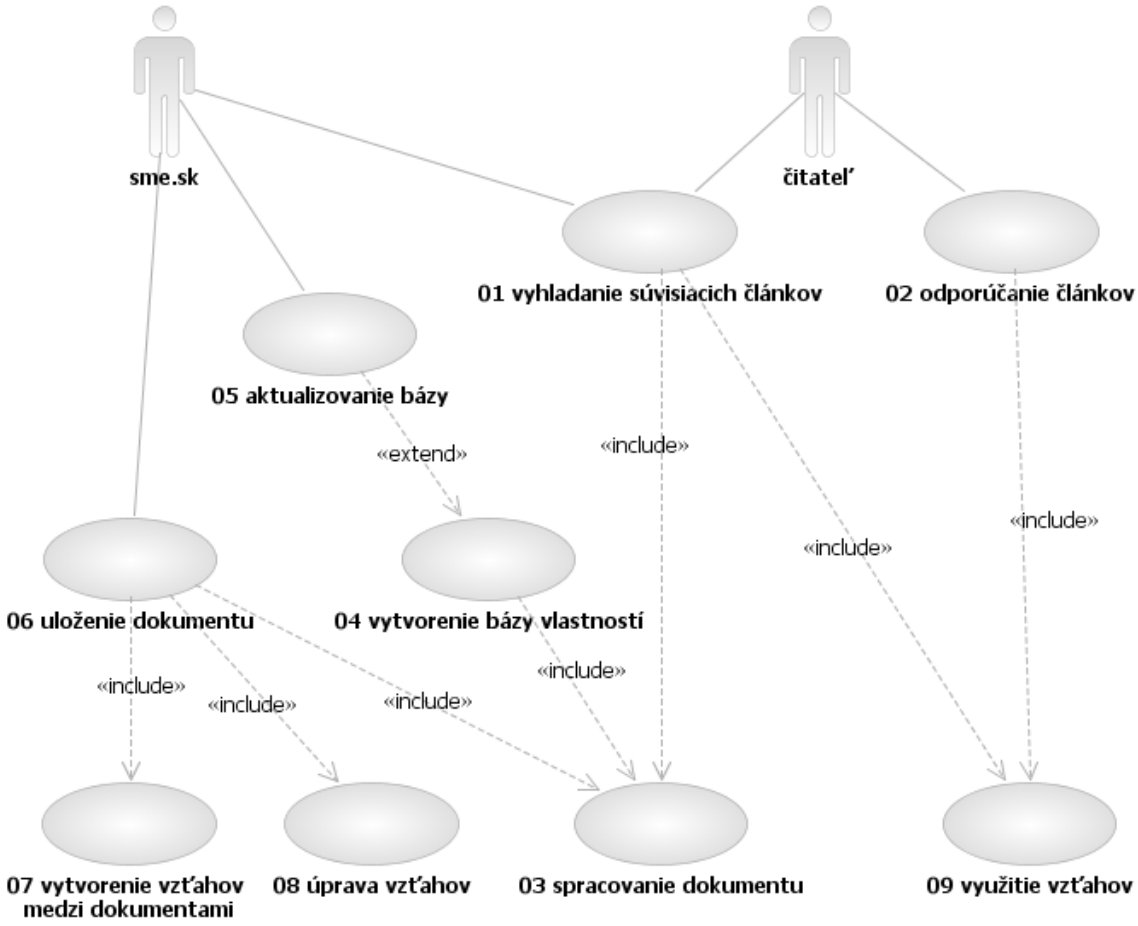
[29] Wen, Y. 2004. Similarity Search in Metric Spaces. University of Waterloo (2004), 64.

[30] Zeleník, D., Bieliková, M. Dynamics in hierarchical news classification. In: Proc. of 4th Workshop on Intelligent and Knowledge Oriented Technologies, WIKT 2009. Herľany, Slovakia (2009), 83-87.

PRÍLOHY

Príloha A - Diagram prípadov použitia
Príloha B - Diagram tried
Príloha C - Ukážka zdrojového kódu
Príloha D - Používateľská príručka
Príloha E - Príklady odporúčaní z experimentov
Príloha F - Obsah elektronického média
Príloha G - Plagát z IIT.SRC '10
Príloha H - Článok zo zborníka pre WIKT '09
Príloha I - Zaslaný článok na konferenciu RecSys '10

A. DIAGRAM PRÍPADOV POUŽITIA



UC01 - Vyhľadanie súvisiacich článkov

- množina najbližších článkov k vybranému článku
- optimálna veľkosť
- vplyv aktuálnosti na podobnosť
- rozšírenie vyhľadania aj pomocou kľúčových slov

UC02 - Odporúčanie článkov

- množina najbližších článkov k záujmom používateľa (prečítané, časové pásmo)
- optimálna veľkosť

UC03 - Spracovanie dokumentu

- text (množina slov)
- lematizér
- báza vlastností
- vlastnosti dokumentu

UC04 - Vytvorenie bázy vlastností

- vzťahy medzi slovami
- výber slov
- štatistiky opakovania
- wordnet

UC05 - Aktualizovanie bázy

- rozšírenie vytvorenia bázy

UC06 - Uloženie dokumentu

- binárny strom
- oddelene stromy podľa kategórie
- listy sú dokumenty
- vrcholy/spoje sú virtuálne dokumenty

UC07 - Vytvorenie vzťahov medzi dokumentmi

- hierarchia podobnosti dokumentov
- vytvorenie z veľkej množiny na začiatku, jednorázovo/inkrementálne

UC08 - Úprava vzťahov

- pri zmene podmienok (zmena váh v strome)
- inkrementálne rozvíjanie množiny dokumentov
- odoberanie prvkov (exspirácia)

UC09 - Využitie vzťahov

- vyhľadanie podobných dokumentov (súvisiace články)
- ďalšie členenie/spájanie zhlukov (vytvorenie optimálnej množiny výsledkov)
- záujem o špecifické dokumenty (časové náhľady, zmena váh)

B. Diagram tried

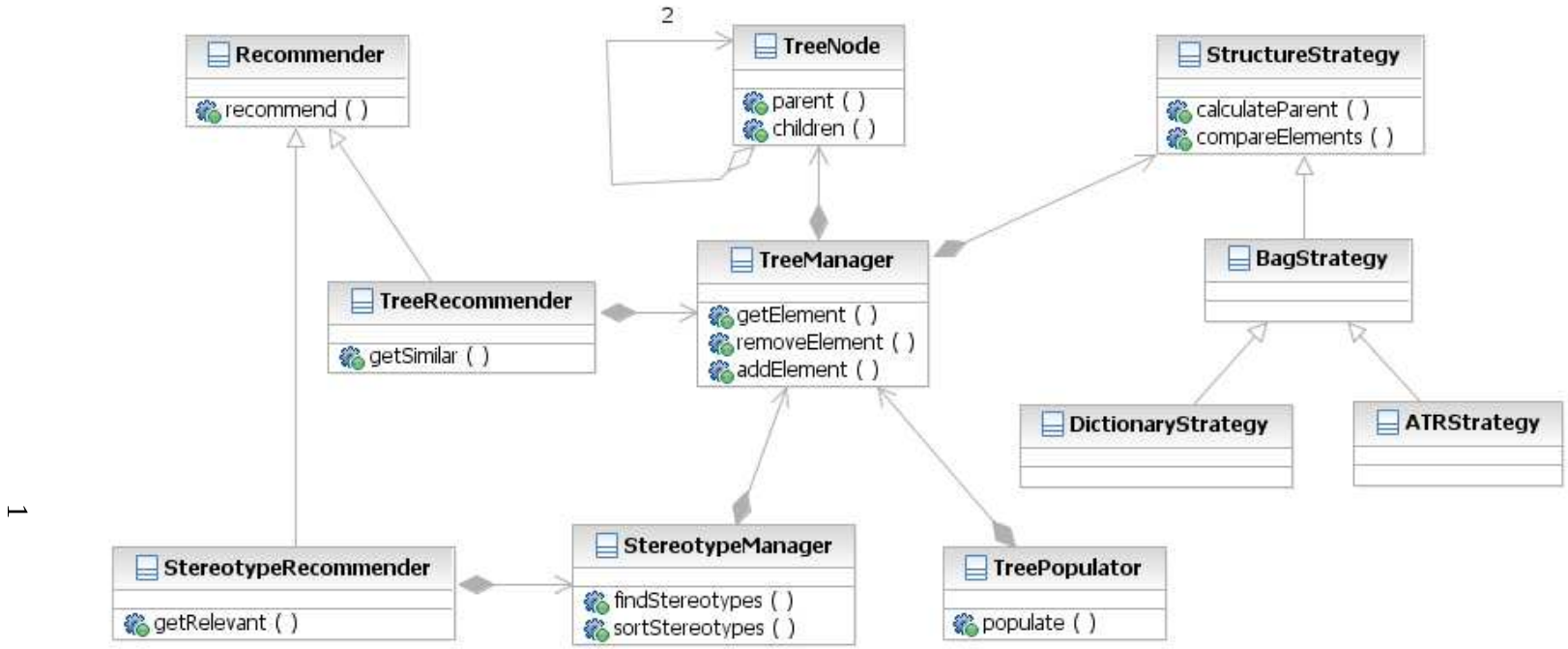


Diagram tried predstavuje náhľad do implementácie riešenia. Ústrednou triedou je TreeManager, ktorý riadi štruktúru stromu a akcie spojené s napĺňaním, modifikáciou a podobne. Zaujímavé je všimnúť si triedy stratégií, ktoré podľa tohto vzoru môžu byť pridávané a tak riešenie výpočtu podobnosti mení svoje vlastnosti podľa tejto implementácie. Významnejšou triedou je aj Recommender, ktorá vo svojich implementáciách slúži na tvorbu odporúčaní v podobne podobných článkov, alebo záujmových článkov. V diagrame nie je zahrnutý celý systém, do ktorého je toto riešenie začlenené nakoľko tento projekt nie je našou vlastnou prácou.

C. Ukážka zdrojového kódu

Šírenie informácií v strome – spojenie vrcholov stromu do metadokumentu

```
#spojenie informácie o hĺbke stromu
def mergeSubTreeDepth(children)
    subtreedepth = 0
    children.each do |child|
        subtreedepth = [subtreedepth, child.subtreedepth].max
    end
    return subtreedepth+1
end

#spojenie informácie o počte potomkov
def mergeChildrenCount(children)
    childrencount = 0
    children.each do |child|
        childrencount+= child.childrencount
    end
    return childrencount
end

#spojenie vlastností (BagOfWordsStrategy)
def mergeFeatures(children)
    answ = {}
    children.each do |child|
        answ.merge!(child.features){|key, new, old| new+old}
    end
    return answ
end

#spojenie informácie o najnovšom článku
def mergeTimeStamp(children)
    newesttimestamp = DateTime.new
    children.each do |child|
        newesttimestamp = child.lastchange if
            child.lastchange > newesttimestamp
    end
    return newesttimestamp
end
```

Výber okolia zo stromu

```
#parameter element je jeden prvok stromu
#parameter minlimit určuje koľko prvkov chceme vybrať
def getMatchingNodeSurround(element, minlimit)

    return nil if element.nil?

    pointer = element
    #vyberiem postupne prechádzam na rodičov, kým nespĺňam podmienku
    while !(parent=pointer.parent).nil? and (parent).childrencount < minlimit
        pointer = parent
    end

    #vrátim množinu, určenú rodičom, plňajúcim podmienku
    return TrecomNode.getLeafSet(pointer)
end
```

Výber podobných článkov

```
#parameter  sme_id  je identifikátor článku
#nepovinný parameter surround určuje okolie článku
def getSimilarArticles(sme_id, surround=500)

    #vyhľadám článok v strome
    element = TrecomNode.getLeafNode(sme_id)

    #skončím, ak nie je v strome
    return nil if element.nil?

    answ = []
    #vyberám okolie elementu
    set = @manager.getMatchingNodeSurround(element, surround)

    #množina sa zoradí podľa podobnosti k prvku
    set.each{|elm|
        similarity = @manager.compare(element, elm)
        answ << [elm.sme_id, similarity] unless
            elm.sme_id == sme_id or similarity == 0
    }
    answ.sort!{|x,y| y[1]<=>x[1] }

    return answ
  end
```

Výber článkov podľa záujmu používateľa

```
#parameter visit hovorí o súčasnej návšteve
def getStereotypeArticles(visit)
    #vyberiem poslednych 50 návštev čitateľa
    visits = Visit.find(:all, :conditions => {:cookie => visit.cookie},
                        :limit => 50, :order => "happened_at DESC")

    #rozdelím množinu na tie, čo boli odporúčané a tie čo našiel sám
    #(každý odporúčaný má logovaný aj redirect - ako keby ho našiel sám)
    recvisits = visits.select{|elm| !elm.recommendation.nil?}
        visits.reject!{|elm|recvisits.each {|recvisit|
        break if recvisit.sme_id == elm.sme_id }
    }
    #v prípade, že čitateľ nič neprečítal odporúčim najnovšie články
    return TrecomNode.getNewestSet(TrecomNode.root, 10) if
        visits.nil? or visits.empty?

    #vyhľadám stereotypy záujmov
    stereotypes = @manager.getStereotypes(visits)
    #vyberiem najrelevantnejšie stereotypy záujmov
    topstereotypes = @manager.getTopNStereotypes(stereotypes)

    #v prípade, že neexistujú relevantné stereotypy odporúčim najnovšie články
    return TrecomNode.getNewestSet(TrecomNode.root, 10) if
        topstereotypes.nil? or topstereotypes.empty?

    answ = nil
    topstereotypes.each do |stereotype|
      #množina už videných článkov
      seen = stereotype.seenset
      #najnovších 10 článkov zo stereotypu, bez videných
      possible = TrecomNode.getNewestSet(stereotype.trecom_node, 10) - seen
      #odmietnem články, ktoré už boli odporúčané
      possible.reject!{|elm| recvisits.each{|recvisit|
          break if recvisit.sme_id == elm.sme_id}
      }

      #množinu pre stereotyp zoradím podľa aktuálnosti
      possible.sort!{|x,y| y.article.published_at<=>x.article.published_at}

      #pridám články do odporúčacej množiny (výsledok bude transponovaný)
      (answ.nil? or answ.empty?)? answ = possible : answ=answ.zip(possible)

    end
    #vrátim množinu unikátnych článkov
    return (answ.flatten-[nil]).uniq

  end
```

D. POUŽÍVATEĽSKÁ PRÍRUČKA

Inštalácia rozšírenia

Celé rozšírenie je momentálne dostupné ako rozšírenie prehliadačov, preto je nutné nainštalovať potrebné súbory, aby sa tak zmeny na stránke http://sme.sk prejavili.
Na stránke http://nimbus.fiit.stuba.sk/sme-recommender/trecom_script/ je možné nájsť aktuálnu experimentálnu verziu, odporúčacieho systému, aj podrobný návod, ktorý sprevádza používateľa pri inštalácií. Potrebné súbory sú uložené aj na elektronickom médiu.
V budúcnosti môže byť inštalácia vypustená v prípade, ak sa riešenie nasadí priamo na portál sme.sk.
Ďalšou možnosťou je inštalácia a používanie proxy servera (peweproxy.fiit.stuba.sk). Ktorá na stránky sme.sk pridáva potrebné prvky.
Inštalácia pre prehliadač FireFox prebieha v krokoch:

1. Nainštalovať plugin Greasemonkey pre FireFox.
2. Nainštalovať skript odporúčacieho systému.

Pre prehliadač Opera prebieha v krokoch:

1. Povoliť Javascript.
2. Nainštalovať opera UserScript nakopírovaním do adresára s ostatnými skriptami.

Prehľad podobných článkov

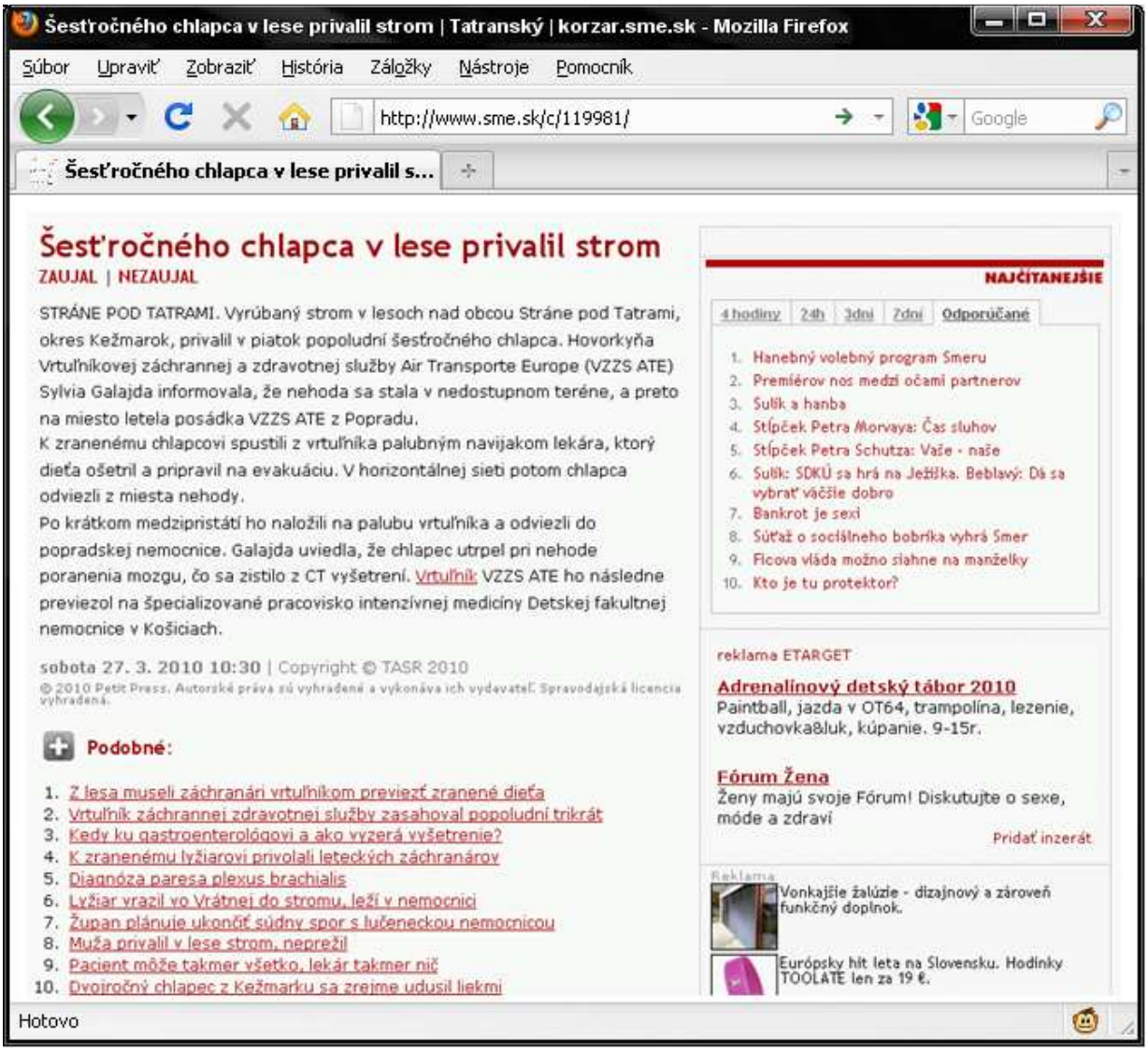
Články podobné k práve čítanému článku sa objavujú pod akýmkoľvek zobrazeným článkom na sme.sk. Kliknutím na niektorý článok so zoznamu sa pre presúvame na ďalší článok. Na demonštračnom obrázku sú podobné články pod článkom v časti "Podobné články".

Prehľad odporúčaní

Odporúčania sa používateľovi zobrazia pri príchode na stránku sme.sk v pravom hornom rohu medzi najčítanejšími článkami pod menom "Odporúčané". Odporúčané články sa obnovujú každú hodinu. Nie je vylúčené, že niektoré odporúčané články v zozname ostanú, ak ich používateľ v poslednej hodine nezobrazil.

Spätná väzba

Po nainštalovaní rozšírenia sa pod nadpisom každého článku objavia dve možnosti, ako ohodnotiť záujem o článok. Tieto sú v podobe textu, alebo obrázku, ktorý vyjadruje záujem. Po kliknutí na možnosť tento prvok zmizne. Prvok sa pri ďalšom načítaní stránky opäť objaví pre prípad zmeny názoru.



Obrazová demoštrancia používateľského rozhrania.

E. PRÍKLADY ODPORÚČANÍ Z EXPERIMENTOV

Experimenty s odporúčaním prebiehali najmä v simulovanej forme podľa aktivít používateľov. Prinášame aj vybrané prípady odporúčaní pre konkrétnych používateľov. Čitateľ je najprv charakterizovaný zoznamom sekcií, ktoré číta a následne sú pre tohto jednotlivca zobrazené aj konkrétne články. Tieto články sú z doby 15. – 20. Apríla.

Používateľ čítajúci blogy
História čítania podľa sekcií na sme.sk:

korzar	**23**
liptov	**6**
komentare	**5**
wiezik	**2**
topercer	**2**
pocitace	**2**
jakus	**1**
barlog	**1**
hajdusek	**1**
marekgajdos	**1**
jurajlukac	**1**

Tento čitateľ okrem všeobecných tém číta aj blogy. Blogy samotné nie sú zaradené do rovnakej sekcie, ale sú navzájom niečím podobné (obsahová podobnosť). Túto podobnosť reprezentuje strom, preto sú blogy iných autorov v zozname odporúčaných článkov (4 z 10).

Neštátni vlastníci lesov nesúhlasia so zonáciou Tatier
Nová zonácia Tatier dáva hotelierom šestinu Manhattanu
Počet obetí zosuvov pôdy v Brazílii stúpol na 219
Posyp znepríjemňuje život Ťahanovčanom
Moja kríza má krízu
Ministerstvo predložilo návrh zonácie Tatranského národného parku
Slovensko zruší pobočku veľvyslanectva v Bonne
Platini a Blatter vyjadrili Poľsku sústrasť a podporu
Odzátkovánie a nalievanie vína.
Ako život plynie..

vypočítané za 2.9375 sekundy pre cookie 12315003471257690

Používateľ čítajúci ekonómiu
História čítania podľa sekcií na sme.sk:

ekonomika	**26**
komentare	**11**
bratislava	**7**
natankuj	**3**
hokej	**2**
veda	**2**
sport	**1**

Tento čitateľ sa venuje prevažne ekonómii. Okrem toho číta komentáre a ostatné témy sú už významovo slabšie pri hľadaní jeho záujmov. V strome túto obsahovú podobnosť článkov o ekonómií vieme lokalizovať. Na úrovni, ktorú sme odhalili potom odporúčame nasledovné články. Vidíme pokrytie článkov z ekonomického prostredia, ale aj články všeobecnejšie. Články pokrývajú aj menej zastúpené sekcie natankuj, veda, hokej a šport.

Športovisko na Budatínskej dokončia asi na etapy
Tadič: Srbsko nikdy neuzná nezávislosť Kosova
V Česku klesla nezamestnanosť na 9,7 percenta
Z emisií sa stratilo ďalších 24 miliónov korún
Fidesz rokoval s rakúskym vicekancelárom Pröllom
Česi vlani vypili menej piva
V Afrike je voľných ešte pol milióna vstupeniek
Transpetrol musí platiť. Česi uznali Ilčišina
Biodiverzitu druhov má zachrániť informačná kampaň
Mikolaj už podpísal zmluvu k zimnému štadiónu

vypočítané za 1.578125 sekundy pre cookie 12475494038518397

Používateľ bez rozpoznateľného záujmu
História čítania podľa sekcií na sme.sk:

korzar	5
hokej	5
nitra	4
prievidza	4
bratislava	4
kysuce	2
trnava	2

Tento čitateľ na prvý pohľad podľa sekcií neprejavuje konkrétne záujmy. Číta prevažne nezaradené články. Podľa článkov z jeho histórie, ktoré sme.sk zaradilo do sekcií vieme identifikovať iba slabú prevahu niektorých sekcií. Hokej ako významnejšia sekcia sa prejavila vo viacerých položkách. Avšak môžeme pozorovať aj článok (4. v poradí) venujúci sa problematike s lokalitou Prievidza. Ďalšie články sú skôr aktuálne doplnenie vo všeobecnej téme.

Slovenskí politici kondolujú Poľsku
ÚPN o Sidorovom antisemitizme pomlčal
Kaczyński - antikomunista, kritik Ruska a silnej únie. A konzervatívec
Minister obrany odvolal riaditeľa opravárenského podniku
Ak Chicago vypadne, Kopecký na MS príde
Biodiverzitu druhov má zachrániť informačná kampaň
Hudáček: Bol to veľmi lacný gól
Kto si Lucifer?
Poľsko prišlo o elitu. Čakajú ho smutné prezidentské voľby
Zmienka o okupácii stranám chýbala, prezidentovi a Smeru nie

vypočítané za 1.15625 sekúnd pre cookie 12634684274107298

Používateľ so záujmom o šport
História čítania podľa sekcií na sme.sk:

korzar	**12**
hokej	**11**
futbal	**11**
ekonomika	**7**
natankuj	**5**
kultura	**4**
cestovanie	**3**
pocitace	**3**

Čitateľ má zjavne záujem o šport, konkrétne futbal a hokej. Okrem týchto tém sa venuje aj ďalším témam, napríklad ekonómií kultúre, cestovaniu. Odporúčané články pokrývajú tieto témy, čo vidíme priamo z nadpisov. V zozname sú články o futbale, hokeji ale aj o kultúre a ďalších.

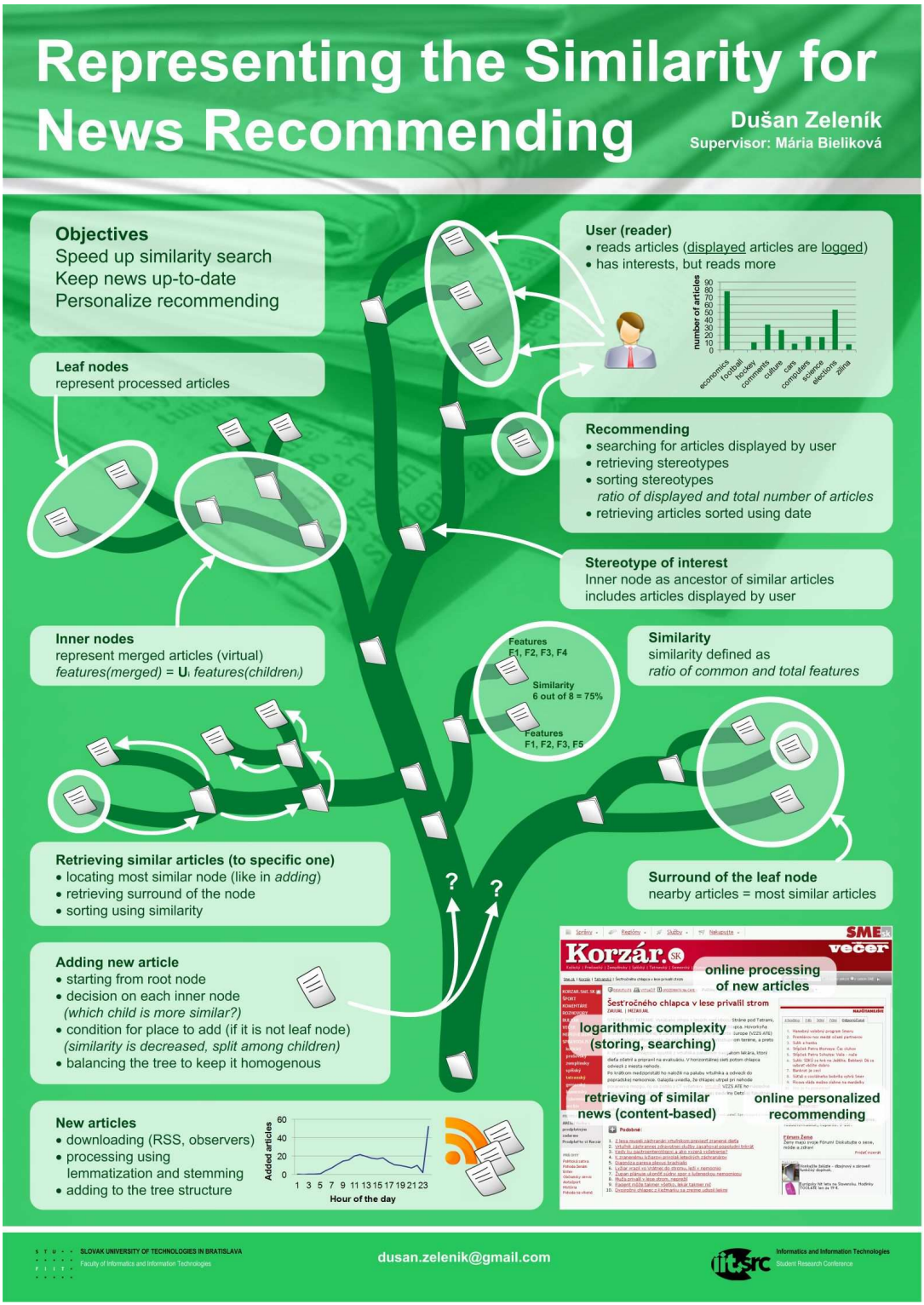
Týždeň v kultúre
Zoltán Varga zomrel na trávniku počas zápasu veteránov
Hodina dvanásta ešte neodbila
Al-Kájda chystá útok počas futbalových majstrovstiev
V Ivangorode podpisujú petíciu za pripojenie mesta k Estónsku
V prvej nominácii Ruska už aj hráči z NHL
John Forsythe - tvár z Dallasu
Jeruzalem - mesto nevyslovených želaní
Messi vyniká i medzi najlepšími
Reprezentačnú pozvánku si nezaslúžim, vyhlásil Ronaldo

vypočítané za 1.984375 sekundy pre cookie 12582136171562522

F. OBSAH ELEKTRONICKÉHO MÉDIA

Priečinok	**Obsah priečinka**
/DOC	dokumenty a prezentačný materiál
/DOC/DP1	diplomový projekt I
/DOC/DP2	diplomový projekt II
/DOC/DP3	diplomový projekt III
/DOC/STEMMING	dokumentácia stemovača
/DOC/WIKT09	materiál k pracovnej dielne
/DOC/IIT.SRC10	materiál k študentskej konferencie
/DOC/RECSYS10	materiál ku konferencii
/DOC/SIGIR10	materiál ku konferencii
/SRC	zdrojové kódy
/SRC/STEMMER	generátor koreňov a analyzator ferretu
/SRC/SMEFIITPROJEKT	ruby on rails aplikácia (integrovaný DP projekt)
/SRC/TRECOM	trecom implementácia DP projektu
/DAT	dátové súbory
/DAT/MYSQL6	spustitelný obraz databázy

G. Plagát z IIT.SRC '10



H. Článok zo zborníka pre WIKT '09

Dynamika v hierarchickej klasifikácii článkov

Dušan Zeleník, Mária Bieliková

Ústav informatiky a softvérového inžinierstva,
Fakulta informatiky a informačných technológií, Slovenská technická univerzita,
Ilkovičova 3, 842 16 Bratislava, Slovensko
dusan.zelenik@gmail.com, bielik@fiit.sk

Abstrakt. Podobnosť článkov môžeme identifikovať aplikovaním klasifikačných algoritmov. Problém nastáva pri neustálej obmene dokumentov a teda nutnosti aktualizovať klasifikačné triedy. Prirodzene články pribúdajú, ale aj starnú a teda opúšťajú množinu. Naša metóda je schopná prispôsobiť sa dynamickej povahe dát využitím hierarchie stromu a menej náročnej aktualizácie vzťahov. Stromová štruktúra rozmiestňuje viacrozmerné prvky podľa podobnosti a dovoľuje generovať množiny zhlukov na základe parametrizovateľných obmedzení. Metóda je použiteľná na získanie súvisiacich článkov alebo tvorbu odporúčaní pre stereotypy používateľov dynamicky v reálnom čase.

Kľúčové slová: dynamika, klasifikácia, odporúčanie, hierarchia

1 Úvod

Významným atribútom pohodlia používateľa je náročnosť cesty, ktorú musí prekonať pri hľadaní informácií. Internetové noviny prichádzajú s možnosťami, ktoré umožňujú čitateľovi v rozsiahlej množine článkov vyhľadať ten pravý. Štandardné metódy spracovania a klasifikácie dokumentov však neuvažujú dynamicky sa meniace podmienky. Čas je pritom v spravodajstve významný atribút. Rovnako aj záujem človeka o konkrétne oblasti informácií.

Cieľom príspevku je opísať metódu odporúčania na základe klasifikácie a vyhľadávania podobných dokumentov, ktorá rieši dynamiku informácií v internetových novín. Metóda je nadstavbou už overených riešení, vložených do meniaceho sa prostredia. Predpokladá sa najmä zlepšenie orientácie v dokumentoch a prispôsobenie sa vkladaniu nových článkov a potlačenie zastaraných. Používateľ získa možnosti ako prehľadávať noviny a ostať v oblasti záujmu bez redundancie.

Klasifikácia objektov je vo všeobecnosti hľadaním funkcie $k\colon d \rightarrow c$ kde d je objekt, v tomto prípade dokument a c je trieda, do ktorej patrí. Problém tohto zobrazenia vzniká, keď sa množina dokumentov stále mení, čo vyžaduje aj zmeny samotnej množiny tried. Spracovanie dokumentov sa najčastejšie rieši využitím množiny slov alebo množiny fráz vybraných na základe štatistiky [6] alebo pomocou vhodnej heuristiky. Problém je, že práca s takto získanými vektormi reprezentujúcimi dokumenty je výpočtovo náročná. Nami použitá nadstavba *TF-IDF*, je schopná určiť štatisticky, ktoré slová sa na vektore podieľajú

viac alebo menej. Využíva váhy slov získané z databázy článkov. Sémantiku textu vieme čiastočne zakomponovať vylúčením nepodstatných slov normalizáciou slov na ich základný tvar a následne "zjednotením" synonym. Slová a ich význam majú však komplikovanejšie vzájomné vzťahy [5]. Niektoré tieto vzťahy sú pre anglický jazyk zachytené napr. v databáze WordNet, ktorá sa s dobrými výsledkami používa na spracovanie textu [1].

Riešením ďalšieho kroku klasifikácie a teda samotného rozdelenia dokumentov, resp. vektorov je vytvorenie zhlukov. Existujú mnohé metódy tvorby zhlukov [3]. Naša metóda, inšpirovaná odvodeninami algoritmu *Leader-Follower*[2], disponuje netriviálnu aktualizáciu týchto zhlukov (nejde iba o zobrazenie nového prvku do už existujúcej množiny centier). Algoritmus aktualizuje priebežne centrá zhlukov. Na vytvorenie hierarchie [4] medzi dokumentmi sme využili štruktúru strom. Tá ak sa vhodne konštruuje umožňuje aj efektívne pridávanie a odoberanie prvkov.

2 Štruktúra a reprezentácia dokumentov a vzťahov

Reprezentácia dokumentov je pre odporúčanie na základe podobnosti kľúčová. Navrhli sme reprezentáciu, kde štruktúra elementov je usporiadaná v strome, ktorého listy predstavujú skutočné dokumenty. Hrany tohto stromu určujú hierarchiu klasifikácie dokumentov. Vrcholy predstavujú elementy s vlastnosťami dokumentu (klasifikačné triedy). Tie udržujú informáciu o svojich potomkoch. Cesta od koreňa stromu do listu je postupnosť tried. Strom je iba úložiskom vzťahov podľa podobnosti dokumentmi, nie samotných dát. Výstupom nie je jeho vizualizácia používateľovi, ale pri tvorbe výstupov sa využívajú hierarchické vzťahy, ktoré sú v ňom zachytené.

Element stromu je odkaz na dokument alebo triedu dokumentov. Každý rodič nesie informáciu o podobnosti s potomkami a vzájomnej podobnosti jeho dvoch potomkov. Tieto vzťahy rozširujú binárny strom o ďalšie užitočné informácie, potrebné pri pokročilom rozhodovaní o ceste k listom pri tvorbe odporúčaní.

Takýto rozšírený binárny strom je výhodným riešením pre pridávanie prvkov. Prechod od koreňa stromu až po list uskutočňujeme binárnym porovnaním potomkov a určením bližšieho elementu. Zložitosť pridávania alebo lokalizovania prvku sa zmenšuje s hustotou stromu a jeho šírkou. V prípade najhlbšieho stromu (najhorší prípad) sa vykoná N operácií rozhodnutia. V najširšom strome a teda v najideálnejšom prípade je to $log_2(N)$ porovnaní (kde N je počet dokumentov).

Veľkosť stromu vieme redukovať rozdelením štruktúry na podstromy (viaceré stromy podľa kategórií, v ktorej sa dokument nachádza). Kategórie vystupujú ako dostupná informácia a konečná množina metadát k použitým dátam.

Strom a celá jeho štruktúra sa vytvára inkrementálnym pridávaním prvkov:

1. nájdi "najbližší" element (list alebo spoj),
2. pred element pridaj nový spoj ako rodiča
3. pridaj potomka ako list do binárneho páru
4. uprav spätne atribúty spojov (vzdialenosti) až po koreň.

Obdobne postupujeme aj pri odstraňovaní prvkov. Prístup sme zvolili najmä pre rastúcu veľkosť množiny. Pri akejkoľvek postupnosti pridávania prvkov je výsledný strom rovnaký. Štruktúra a udržanie vzťahov medzi jednotlivými elementmi sú dôležité práve pre ďalšiu prácu s nimi.

Prierezom stromu (medzi listami a koreňom) vieme získať množinu elementov oddelených rezom. Všetky získané elementy sú koreňmi nových stromov. Elementy môžu mať svojich potomkov a teda vieme určiť skupiny podobných dokumentov ako množiny listov vzniknutých stromov. Ak hýbeme rezom smerom ku koreňu, jemnosť skupín sa znižuje (vzniká menej skupín a viac dokumentov v jednej skupine). Rezmi bližšími ku listom stromu sa prejaví opačný efekt zvyšovania jemnosti. Výhodou je postupné rozdeľovanie dokumentov. Menej podobné dokumenty sa oddelia skôr ako tie blízke. Dosahujeme tak generovanie zhlukov zmenou parametrov ako je ich počet alebo veľkosť zhluku.

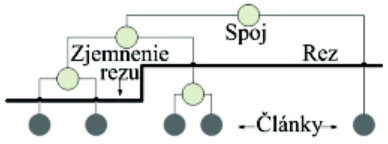
3 Dynamika vzťahov medzi dokumentmi

Navrhnutý strom udržuje informáciu o podobnosti dokumentov. Pri hľadaní v strome alebo tvorbe skupín vieme zohľadniť aj čas, záujem používateľa alebo kľúčové slová. Od koreňa až po list prechádza algoritmus cez spoje. Ak sa zmenia vlastnosti spojov (váhy slov bázy, zohľadnenie času), vyhľadá inú cestu k výslednému listu. Vieme ovplyvniť vzájomnú vzdialenosť dvoch potomkov (horizontálny zásah) alebo vzdialenosť rodiča od potomkov (vertikálny zásah). Horizontálny zásah je posunutie spoja (a jeho rodičov) bližšie k dokumentu, ktorý je napríklad novší. Vertikálnym zásahom docielime zrýchlenie alebo spomalenie zjemňovania pri vytváraní rezu na konkrétnych vetvách (tvorba zhlukov).

Listy obsahujú informáciu, kedy boli pridané. Túto informáciu so zvyšujúcou sa váhou smerom k listom využívame pri rozhodovaní zaradenia vyšetrovaného článku k jeho najbližšiemu. Z toho vyplýva možnosť vzniku situácie, keď kópia vyšetrovaného článku bola zaradená v strome už dávno. Vtedy algoritmus vyhľadá menej podobný, ale aktuálnejší článok. Článok sa takýmto zásahom postupne dostáva za prah, kedy ho možno vylúčiť, čím sa strom aktualizuje.

V niektorých prípadoch (napr. horizontálny zásah) možno vykonať požiadavku priamo nad stromom. Vzniká rez (pozri obr. 1), ktorý určuje jemnosť delenia na konkrétnych vetvách stromu (napríklad podľa záujmu).

Spoje možno rozšíriť aj o ďalšie informácie ako väzby na typy používateľov – miesto pre používateľa v strome určujúce jeho záujmy.



Obr. 1. Zjemnenie rezu (pozn.: hrany iba vizualizujú vlastnosti spojov)

4 Vstupné dáta a ich spracovanie

Metadáta (autor, kategória článku) nie sú dostatočné ukazovatele. Vytvárame preto vektor obsahujúci zložky reprezentujúce frekvencie výskytov slov. Tvorba takého vektora využije zobrazenie foriem slov na ich základnú formu. Tým sme schopní jeho veľkosť zredukovať, avšak pri počte základných foriem slovenského jazyka (vyše 50 tis.) je počet zložiek vektora náročný pre časté porovnávanie.

Metóda vyberá špecifické slová výhradne pre danú doménu (noviny). Ich počet sa redukuje uprednostnením slov, ktoré určujú najlepšie význam samotného článku. Technika výberu slov je založená na štatistike (používame dostatočne veľkú databázu článkov). Navrhnutá metóda takto zohľadní aj vzťahy medzi slovami podľa ich vzájomnej nadradenosti (priority) v tejto doméne. Vysoké počty výskytov jednotlivých slov znižujú ich relevantnosť, rovnako však aj slová so zanedbateľným výskytom môžeme považovať za irelevantné. Zohľadňujeme aj rôznorodosť ich výskytu v dokumentoch. Slová s rovnakou frekvenciou vo všetkých dokumentoch sú menej dôležité, ako tie s rôznou.

Určenie vektora pre dokument je dôležité pre ich porovnanie, tak ako aj spôsob porovnania. Pre porovnanie dokumentov sme použili metriku založenú na kosínovej podobnosti. Tá zanedbáva veľkosť vektora, využíva jeho smer.

Bázou slov rozumieme množinu takto získaných slov. Vytvorenie bázy je jednorazová záležitosť (objavenie vlastností dokumentov). Aktualizácia bázy má význam po pridaní nových článkov. Takto postupujeme len ak je báza zastaraná (zastarávanie bázy však v špecifickej doméne nie je kritické).

Bázu slov sme získali štatistickým vyhľadaním slov, ktoré majú predpoklad, že určujú črty dokumentu. Tieto slová sme potom zoradili podľa dôležitosti. Na základe týchto slov potom môžeme generovať vektory ku ďalším článkom. Algoritmus postupuje takto:

1. stiahnutie článku s metadátami (text, názov, kategória, autor, čas pridania),
2. normovanie textu (preklad foriem na lémy a vylúčenie "obyčajných" slov),
3. získanie bázy slov b (každé slovo b_i získa váhu v_i podľa výskytu v jednotlivých článkoch a výskytu vo všetkých článkoch)
   - váha je nízka ak je frekvencia výskytu slova vo všetkých článkoch rovnaká
   - váha sa logaritmicky zvyšuje s rôznorodosťou výskytu,
4. zmenšenie bázy slov o slová s významne nižšou váhou.

Algoritmus pre vytvorenie vektora ohodnocujúceho článok pracuje s bázou slov. Slová bázy majú svoje váhy a teda priority, na základe ktorých sa môžu tvoriť vektory s ohodnotenými (zoradenými) zložkami. Zložky majú rôznu dôležitosť (váhu). Algoritmus priraďujúci vektor článku prebieha v týchto krokoch.

1. stiahnutie článku s metadátami (text, názov, kategória, autor, čas pridania)
2. normovanie textu (preklad foriem na lémy a vylúčenie "obyčajných" slov)
3. vyhľadanie slov z textu v báze a vygenerovanie vektora $v = (v_1, v_2, v_3....v_n)$ kde v_i je frekvencia výskytu slova b_i z bázy b v normovanom texte.

Frekvencia výskytu je kalkulovaná ako *TF-IDF*. Ak chceme rozšíriť stratégiu pomocou miery významu slov v doméne (rozptyl, maximálny a priemerný výskyt slova), využijeme projekciu vektora slov v. Takto získajú zložky vektora svoje váhy a jeho smer sa blíži k najvýznamnejšiemu slovu.

Ďalšou stratégiou je využitie synoným a teda priemetu slov rovnakého významu na určené slovo (napr. prvé v synonymickom slovníku). Jedná sa o významne zníženie počtu zložiek vektora a teda aj náročnosti porovnaní.

5 Overenie riešenia a záver

Najdôležitejším kritériom overenia správnosti navrhnutej metódy je spokojnosť používateľa. Noviny disponujú záznamami o činnosti používateľov a tak vieme určiť, ktorý článok jednotlivca zaujal. Záznamy sú upravené, aby bolo zrejmé akým spôsobom sa používateľ dostal k článku. Cieľom metódy je poskytnúť používateľovi zoznam pre neho zaujímavých článkov. Ak zoznam začne využívať pravidelne pre hľadanie článkov, čo je pozorovateľné, potom je metóda úspešná.

Zo záznamov je tiež možné zistiť, či sa články prečítané používateľom náhodné rozptyľujú v strome tried alebo sa s malou odchýlkou približujú konkrétnym oblastiam. V tom prípade to znamená, že používateľ nestráca čas zobrazovaním nezaujímavých článkov. Naopak články, o ktoré má záujem si zobrazí. Rozdiel pozorujeme porovnaním záznamov o používateľovi pred a po aplikovaní riešenia.

Pri pozorovaní je vhodné určiť viac dostatočne veľkých skupín reprezentatívnych používateľov. Jedna skupina používa pôvodný systém. Ďalšia skupina používa rozšírenia. Obe skupiny sa nakoniec porovnajú a určíme zlepšenie. Úspešnosť vieme porovnať s ďalšími metódami pre odporúčanie článkov.

Pomocou navrhnutej metódy podávame výstupy vo forme množín podobných článkov a generovaní odporúčaní vzhľadom na záujmy používateľa. Významným príspevkom je zvládnutie dynamiky novín, ktorá ovplyvňuje výstupy metódy.

Poďakovanie. Táto práca bola čiastočne podporená Vedeckou grantovou agentúrou VEGA, grant VG1/0508/09 a Agentúrou na podporu výskumu a vývoja, grant APVV-03961-06.

Literatúra

1. Choi Ben, Peng Xiaogang.: *Document Classifications Based On Word Semantic Hierarchies*. Computer Science, Louisiana Tech University. 2005.
2. Duda, R. O., Hart, P. E., and Stork, D. G.: *Pattern Classification*. 2nd Edition. Wiley-Interscience. 2000.
3. Jain, A.K, Murty, M.N. and Flynn, P.J.: Data Clustering: A Review. In *ACM Comp. Surveys*. Vol. 31, No. 3, 1999, 264-323.
4. Şerban, G., Câmpan, A.: Hierarchical Adaptive Clustering. In *Informatica*, Vol. 19, No. 1, 2008, 101-112.
5. Vojtek, P., Bieliková, M.: Homophily of Neighborhood in Graph Relational Classifier. In *Lecture Notes in Computer Science*, Proc. of SOFSEM 2010. To appear.
6. Zhong, S., Ghosh, J.: Generative model-based document clustering: a comparative study. In *Knowl. Inf. Syst.*, Vol. 8, No. 3, 2005, 374-384.

I. ZASLANÝ ČLÁNOK NA KONFERENCIU RECSYS '10

News Recommending Based on Similarity Relations

Dušan Zeleník
Institute of Informatics and Software Engineering
Faculty of Informatics and Information Technologies
Slovak University of Technology
Ilkovičova 3, 842 16 Bratislava, Slovakia
dusan.zelenik@gmail.com

Mária Bieliková
Institute of Informatics and Software Engineering
Faculty of Informatics and Information Technologies
Slovak University of Technology
Ilkovičova 3, 842 16 Bratislava, Slovakia
bielik@fiit.stuba.sk

ABSTRACT
This paper discusses qualities of the hierarchical representation of text documents and its utilization for the news recommending. We propose a method for efficient representation of entities in the hierarchy and present how this method maintains similarity relations. We use this representation to provide content-based recommendations for news readers. Our solution provides recommendations, while solving known drawbacks mentioned in related work. Both parts, representation and recommender, were evaluated to verify partial potentials. We used dataset of articles and history of the active readers to evaluate recommender in simulated usage.

Categories and Subject Descriptors
H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval|*RetrievalModels*

General Terms
Algorithms, Design, Experimentation

Keywords
Similarity search, tree hierarchy, news recommending

1. INTRODUCTION AND RELATED WORK
Generating personalized recommendations is generally related to observation of a single user behavior and then suggesting items using either collaborative filtering or content-based methods. In advance there are also hybrid techniques, which could be used to recommend items [4]. These methods often combine both principles to avoid negative aspects in both types.

Collaborative filtering is purely social based what commonly emerges in recommending the most popular items. Ignoring the content of entities and focusing on the similarity of users is based on presumption that the majority of similar users have found something interesting for the rest. Actually this is more about predicting the behavior than discovering the interest. More on collaborative filtering recommenders could be found in the survey [9], since we focused on the content-based recommending.

Recommender systems using content of the items have to represent similarity among entities which are subject to recommend. This is the first problem, because in more complex domains it is not, or hardly possible to find similarities. Items like videos or images have none content similarity without keywords or annotation as proclaimed in work on Google News recommender [6]. If the recommender works with news articles, simple text needs to be processed and similarity among documents retrieved [10]. Similarity itself is then used to provide content-based recommendations. The simplest possible way is to recommend articles which are similar to those which the reader has already seen. Problems which occur in content-based recommenders are similar to the collaborative recommenders [1]. There is always a problem with unknown user, but in content-based system we have no problem with new items, in case we can process them and represent similarities. Problem could cause overspecialization, which could be solved by randomly generated recommendations and omitting the items which are very similar to those which the reader already saw like in DailyLearner [3].

As a part of SMEFIIT [1] project we present a recommender called TRECOM which works with monitored user behavior and processed news articles in the form of the effective representation of the news similarity. Data used to perform recommending are from the web site SME.sk and contains news texts and the user history (news reading). History is purely from the news reading, but the intention is similar to the one presentment in related work [5]. We also use this log to experimentally evaluate our recommender by simulating user activity and also via publishing the recommender at the site.

We process articles to provide similarity search service in used dataset. Our method incrementally composes a hierarchy of similarities among supplied text documents. Representation is inspired by incremental hierarchical clustering methods and effectiveness which is achieved in related work. Similarity search is then low complex and effective for discovering interests using history of the specific user. Discovered interests are then in dependency to the time and relevance used to recommend.

2. HIERARCHY OF SIMILARITIES
We use the similarity as an option to search for entities in a big dataset, rather than keywords or categories. We use a technique which is based on similarities comparison. Similarity search mainly depends on the extraction of features for each entity and metric chosen to determine similarity between two entities. Talking about news, mentioned features represent words. Similarity between texts is then computed using sets of words. Computed similarities are important to assign relations between news and to speed up similarity search.

Our representation which keeps similarities among news provides easy real-time adding of new articles and furthermore, retrieving group of similar news. Our method for similarity search is designed to work with any entity type (i.e. pictures, persons), but in our boundaries we focused on the text documents. In this section we describe data preprocessing, similarity calculation and representing relations in hierarchy.

2.1 News attributes extraction

We used a dataset of news articles published with the specific meta-data (title, author, section, category etc.). All attributes could be used to determine whether two or more entities are similar or not. We presume that similarity is not categorical and more articles from different categories could have similar plot what makes this manual categorization insufficient. This insufficiency led us to use the content of each article. Attributes are consequently extracted only from topic, perex and text. Words should be more useful when the plot similarity is important.

The content is preprocessed using the lemmatization, followed by the stemming algorithm to extract words. After tokenizing the article, we apply our dictionary to put words on their base forms. The lemmatizing dictionary is supported by computed stems of words. We produced this dictionary to preserve words like names, brands, acronyms and other valuable word forms, not included in the morphology dictionary. The combination of the official morphology dictionary and stems generated statistically should increase similarity especially for mentioned preserved words.

2.2 Similarity relation

Using features we compute the similarity between two articles. We chose the simple bag of words strategy because of its low complexity and relatively high precision [7]. Our method for similarity search is not dependant on similarity calculation used. The metric has an impact on results and relations between entities [10], but is not essential in the scope of representing computed similarity and representing relations between texts.

Our strategy of comparing articles uses extracted features to calculate the similarity. We calculate the similarity basically as:

$$similarity(A,B) = \frac{|features(A) \cap features(B)|}{|features(A) \cup features(B)|}$$

The similarity between more documents reveals relations. Some of the articles have greater similarity than others, what affects these relations. We can imagine a graph of articles (commonly represented as a matrix) generated according to their relations. This representation provides precise relations map but problem is its complexity in large datasets. We proposed to informally project these relations into hierarchy. In such hierarchy are similar articles close to each other.

2.3 Hierarchy of relations

As far as we are able to calculate the similarity between articles, we are able to compose a structure to represent relations. Our representation is a binary tree. We are inspired by the IHC [8] algorithm which produces a hierarchy of clusters. Our solution strictly produces hierarchy of entities and relations between them. Our binary tree is then used like a decision tree, which means that two nodes are compared when article is being stored or retrieved on each level. Each leaf node in the tree represents a real document (article, picture, etc.) and the rest of nodes are virtual documents. The virtual document acts as a real document, but it is not the regular readable article, presented e.g. in the news portal. Virtual documents have only representational purpose and provide information about leaf nodes which are located in a subtree defined by this document (see Fig. 1).

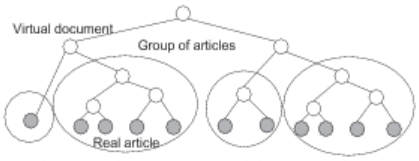


Figure 1. Virtual document joins real articles.

Since the structure is generated using calculated similarity, similar articles are located nearby. Articles could be inserted or retrieved from this structure. The hierarchy enables us to execute these actions with low complexity, because the decision process logarithmically eliminates unnecessary branches.

2.3.1 Inserting news

The tree hierarchy is composed incrementally. The insertion process searches for an insertion place of the new document. Single decision is made according to the defined strategy of similarity metric. The insertion consists of these steps:

1. Searching for the most similar node.
2. Calculating place to insert.
3. Distributing changes.

The process of searching for the most similar node uses extracted features. It starts with the root node and compares its features with children's features. More similar node is then returned for next iteration. Calculated similarity among the new element and the matching node increases after each iteration. Cycle continues until real document is found or features are significantly split between children and similarity declines. These states mean that the algorithm reached the most appropriate virtual document. Features are split into two branches when a set of articles (leaf nodes of the selected branch) is similar to the investigated one and process found the most appropriate place for the insertion.

Insertion continues after calculating the exact place to add the document. The place is calculated to keep the tree well balanced. The homogeneity of the tree is preserved by placing new node without enlarging the tree depth. New document is inserted in the position closest to the most similar node (in the closest and the shortest branch). For example, a set of N documents, which are generally not similar, would be spread into the tree with depth of log2N. This happens more probably in early stages.

After branching at the calculated place, the last stage of inserting the node is executed. The document and its features are distributed. This affects only the parts of the tree which are related to the new node. For each node E basically applies:

$$features(E) = \bigcup_i features(children(E)_i)$$

2.3.2 Retrieving similar news

The location of the article which is the most similar to the investigated one reveals its surroundings. The structure was composed using similarities and relations are hierarchically merging similar articles. The most similar articles are those which

are located nearby. Whole process of similar news retrieving consists of (i) searching for the most similar node and (ii) fetching the set of nearby documents.

Searching for a node, which is the most similar, is the same as the one used for inserting a new document. However, the process firstly checks whether the structure already contains investigated element. In this case, we locate the node and move towards the next step. Nearby documents are more relevant than farther elements but we are able to fetch as many articles as it is necessary in both cases. The greater the surround, the greater is the set of similar articles.

3. RECOMMENDIG USING SIMILARITY

Generating recommendations using representation of similarity relations is based on structure of the tree. We prepared the tree which includes every newly added article and enables us to retrieve similar news. Retrieving similar documents is the simple version of recommender. Suggested articles are the most similar to the one just displayed by a user.

Moving to the next level, fetching similar articles is no more an issue. Our recommending method does not need to fetch similar articles, but uses this tree to locate user interests. We presume that user reads articles, which are somehow similar. Articles are grouped by the similarity what enables to determine which group covers which interest. These interests are then located in the tree structure and recommendation could be prepared. Our recommender uses content-based similarity of articles which are interesting for a user [1]. We presume that users read articles which are included in their interest stereotypes. One stereotype is connected to the specific content of articles. The group of similar articles is represented as a set of leaf nodes in a subtree defined by a virtual document. We use these documents to mark interest stereotypes of the user. Every newly added leaf node is potentially interesting for the user if it is in such a subtree.

To cover all of the user interests, we need to produce a mix of articles sorted by relevance and publishing date. We constrained the number of recommendations to 10, because it would not overwhelm the reader. Modeling the user, picking up the best set of articles and other processes required to generate recommendation are described in this section.

3.1 User model

Using the history of news reading, we discovered that a user has more interests which should be mapped. History of the user contains only logs of displayed articles. We are not able to determine whether the specific article is interesting for the user or not. But using the complete history, we are able to identify trends and simplified user interests (Fig. 2). The chart contains statistics of a specific user for period of one week. Since there are around 200 sections available we are able to recognize interests. Probably, some of the articles displayed by this user were not interesting for the user, but the majority shows his interest stereotypes. Our hierarchy does not contain information about these sections, which were created manually by authors. Our hierarchy keeps computed categories which could not be defined by human, but these categories probably cover (or even correct) sections provided by authors. We presume that the majority of articles displayed by the user are relevant. Discovering interest stereotypes is designed to be able to sort stereotypes by relevance.

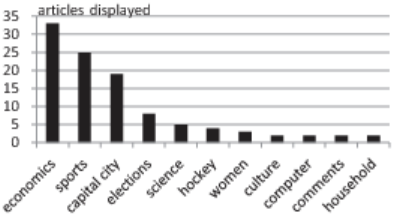


Figure 2. Interests of the specific user.

Mentioned chart simplifies the spectrum of interest which could be obtained using our representation. If the user had an option to decide whether the article was interesting for him or not, we would be able to create better user model. That means that feedback would have significantly affected recommendations. Our intention is to provide recommendations with no feedback required (except experiments), by calculating the relevance of retrieved interests.

3.2 Retrieving interests

If the user likes specific theme, he would like to read the most recent news in this field (e.g. computers). We also presume that user displayed a lot of dated articles from this theme. Our representation provides groups of similar articles on different levels of similarities. Such group is actually represented by an inner node, which merges all of leaf nodes. Stereotype itself is easily gained using the set of articles which are interesting for the user. These articles are located in the tree so we find a path from the leaf nodes to the root. Each intersection which is discovered is the potential stereotype (see Fig. 3).

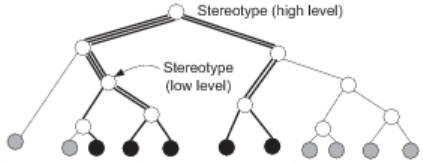


Figure 3. Paths to displayed news and discovered stereotypes.

Each stereotype could be on different level of the tree hierarchy. We calculate a priority of the stereotype as a ratio of total number of leaf nodes in this branch and number of nodes which represent articles already displayed. We can sort stereotypes by priority this way and recommend newly added or unread articles only for top rated stereotypes.

3.3 Generating recommendations

The last step is the composition of the recommendation. Discovered interest stereotypes are used to retrieve articles. Obviously there are more interests and possible recommendations. Average reader displays 2 articles per day, what in comparison to 300 articles published each day means, that there are many relevant articles. The reader should not be overwhelmed by recommendations, what lead us to pick up top 10 stereotypes and for each stereotype we select the newest article. Newest articles are also easily retrieved since each node of the representation has information about the publishing date of the newest article which belongs to the branch. To complete the task of composing the recommendation, we subtract the set of already displayed articles.

Since we included also lower rated, we manage to preserve the negative aspect when recommendations keeps user within limited field of interests.

4. Experiments and results

In order to evaluate reliability of our representation and subsequently the recommender properties we have provided several. Both presented experiments are made using dataset of articles and readers history downloaded from the SME.sk news portal. Complexity, which makes our representation suitable for recommender was evaluated by repeating the process of news processing. Complexity showed to be logarithmical as expected. With linear amount of articles processed the time needed to insert a new article was increasing logarithmically.

We consider proving reliability of proposed representation as more important. To evaluate our presumption that similar articles are located close to each other in the tree structure we compared our method for similarity search with brute force method. We made a comparison of top 10 articles gained by both methods (see Tab. 1) regarding 100 % precision of the brute force method.

Table 1. Precision for top 10 articles. Enlarging size of group.

Surround	10	50	100	200	300	400	500
Precision	27.3	42.8	59.6	78	85.4	90.6	97

This comparison brings valuable results since we compare our results with results computed using high complex brute force method (generally 100 times slower than our method). We used 5 000 articles and we incrementally enlarged the group of similar articles which were retrieved from the tree. Precision is highly affected as a result of the tree composing principle. Similarity relations are generally not transitive, which means that group of similar articles produced by our method includes articles which are highly similar at first place.

Another evaluation proves our hypothesis that our recommender is able to cover user interests. We prepared a set of 1 000 active users and their history for 5 days. We also processed 12 thousands articles displayed by these users. The idea of this experiment is to simulate recommendation using the first period and compare it with the following test period, unknown for the recommender. There is no point to compare exact articles, which were suggested and which were really displayed by the user. This kind of the evaluation would prove the quality of prediction, but not the quality of recommendation. We already showed that user is interested in some specific sections. Furthermore, our dataset also contains categories. The number of unique combinations (section-category) which are also valid is 427 (30 articles per combination in average). With a maximal number of 10 recommended items we can compare combinations displayed by the user and combinations which would be offered by the recommender. This evaluation shows ability to cover the catalogue search and user interests in section-category combination. Our recommender does not use these combinations (it uses content similarity only) and picks maximum of 10 unique combinations (out of 427 valid combinations) what makes it relatively precise (tab. 2).

Table 2. Precision and recall for the changing test period.

Test period	1 hour	4 hours	10 hours	24 hours	48 hours
Precision	40	49	56	58	59
Recall	71	60	44	32	25

5. Conclusions

We presented a recommender which uses history of a single reader and suggests articles using effective similarity search. At the beginning we presented the method for similarity search in the dataset of articles. The main contribution of this solution is its low (logarithmical) complexity. We took advantages of the hierarchical clustering and simplified the processes of the tree composing. Our algorithm incrementally composes a hierarchy of similarity relations and is designed to search for interests using history of the reader. Beyond the ability to locate interests, we are also able to produce hierarchy of these interests and determine relevancy for the user. These operations are executed real-time what enables the user to read interesting and up-to-date articles.

Similarity search results which are achieved are sufficient in comparison with simple brute force method for similarity search. This also showed to be reliable when our method for news recommending suggested articles to users whose behavior was simulated using real logs. Our recommender is ready to accept different types of entities what opens opportunities for future work. Besides, it is possible to implement different metric of similarity calculation. Other challenge for future work is the combination of collaborative filtering and content based approach.

6. References

[1] Adomavicius, G. and Tuzhilin, A. 2005. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. IEEE Trans. on Knowl. and Data Eng. 17, 6 (Jun. 2005), 734-749.

[2] Barla, M., Kompan, M., Suchal, J., Vojtek, P., Zeleník, D., Bieliková, M., 2010. News recommendation. In Proc. of the 9th Znalosti, pp. 171-174.

[3] Billsus, D., Pazzani, M. 2000. User Modeling for Adaptive News Access. User Modeling and User-Adapted Interaction, vol. 10, nos. 2-3, pp. 147-180.

[4] Burke, R. 2002. Hybrid Recommender Systems: Survey and Experiments. User Modeling and User-Adapted Interaction 12, 4 (Nov. 2002), 331-370.

[5] Carvalho, C., Jorge, A. M., and Soares, C. 2006. Personalization of E-newsletters Based on Web Log Analysis and Clustering. In Proc. of the IEEE/WIC/ACM int. Conf. on Web intelligence. IEEE Computer Society, WDC, 724-727.

[6] Das, A. S., Datar, M., Garg, A., and Rajaram, S. 2007. Google news personalization: scalable online collaborative filtering. In Proc. of the 16th int. Conf. on World Wide Web. WWW '07. ACM, NY, 271-280.

[7] Kroha, P.,Baeza-Yates, R., 2005. News classification based on term frequency. In Database and Exp. Sys. Apps., 2005. Proc. 16th Int. Work., pages 428–432.

[8] Sahoo, N., Callan, J., Krishnan, R., Duncan, G., Padman, R. 2006. Incremental hierarchical clustering of text documents. In CIKM '06: Proc. of the 15th ACM int. conf. on Information and knowledge management, NY, USA.

[9] Su, X. and Khoshgoftaar, T. M. 2009. A survey of collaborative filtering techniques. Adv. in Artif. Intell. 2009 (Jan. 2009), 2-2.

[10] Tintarev, N., Masthoff, J. 2006. Similarity for news recommender systems. In Proc. of the AH'06 Work. on Recommender Sys. and Int. User Interfaces.