Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
FIIT- 5208-52497

Bc. Tomáš Chovaňák

ROZPOZNÁVANIE VZOROV SPRÁVANIA POUŽÍVATEĽOV WEBOVÉHO SÍDLA

Diplomová práca

Študijný program:	Informačné systémy
Študijný odbor:	9.2.6 Informačné systémy
Miesto vypracovania:	Ústav informatiky a softvérového inžinierstva, FIIT STU
Vedúci práce:	Ing. Ondrej Kaššák

máj 2017

STU FIIT
SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ

Zadanie diplomovej práce

Meno študenta: **Bc. Tomáš Chovaňák**

Študijný program: Informačné systémy

Študijný odbor: Informačné systémy

Názov práce: **Rozpoznávanie vzorov správania používateľov webového sídla**

Samostatnou výskumnou a vývojovou činnosťou v rámci predmetov Diplomový projekt I, II, III vypracujte diplomovú prácu na tému, vyjadrenú vyššie uvedeným názvom tak, aby ste dosiahli tieto ciele:

Všeobecný cieľ:

Vypracovaním diplomovej práce preukážte, ako ste si osvojili metódy a postupy riešenia relatívne rozsiahlych projektov, schopnosť samostatne a tvorivo riešiť zložité úlohy aj výskumného charakteru v súlade so súčasnými metódami a postupmi študovaného odboru využívanými v príslušnej oblasti a schopnosť samostatne, tvorivo a kriticky pristupovať k analýze možných riešení a k tvorbe modelov.

Špecifický cieľ:

Vytvorte riešenie zodpovedajúce návrhu textu zadania, ktorý je prílohou tohto zadania. Návrh bližšie opisuje tému vyjadrenú názvom. Tento opis je záväzný, má však rámcový charakter, aby vznikol dostatočný priestor pre Vašu tvorivosť.

Riaďte sa pokynmi Vášho vedúceho.

Pokiaľ v priebehu riešenia, opierajúc sa o hlbšie poznanie súčasného stavu v príslušnej oblasti, alebo o priebežné výsledky Vášho riešenia, alebo o iné závažné skutočnosti, dospejete spoločne s Vaším vedúcim k presvedčeniu, že niečo v texte zadania a/alebo v názve by sa malo zmeniť, navrhnite zmenu. Zmena je spravidla možná len pri dosiahnutí kontrolného bodu.

Miesto vypracovania: Ústav informatiky a softvérového inžinierstva, FIIT STU Bratislava

Vedúci práce: **Ing. Ondrej Kaššák**

Termíny odovzdania:

Podľa harmonogramu štúdia platného pre semester, v ktorom máte príslušný predmet (Diplomový projekt I, II, III) absolvovať podľa Vášho študijného plánu

Predmety odovzdania:

V každom predmete dokument podľa pokynov na www.fiit.stuba.sk v časti: home > Informácie o > štúdiu > organizácia štúdia > diplomový projekt.

V Bratislave dňa 15. 2. 2016

prof. Ing. Pavol Návrat, PhD.
riaditeľ Ústavu informatiky a softvérového inžinierstva

STU FIIT
SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ

Návrh zadania diplomovej práce

Finálna verzia do diplomovej práce [1]

Študent:

Meno, priezvisko, tituly:	Tomáš Chovaňák, Bc.
Študijný program:	Informačné systémy
Kontakt:	tchovanak@gmail.com

Výskumník:

Meno, priezvisko, tituly:	Ondrej Kaššák, Ing.

Projekt:

Názov:	Rozpoznávanie vzorov správania používateľov webového sídla
Názov v angličtine:	Recognition of Web user's behavioral patterns
Miesto vypracovania:	Ústav informatiky a softvérového inžinierstva, FIIT STU
Oblasť problematiky:	Personalizácia Webu, Analýza vzorov správania človeka vo webovom sídle

Text návrhu zadania[2]

Vzory správania používateľov vo webovom sídle vyjadrujú opakujúce sa akcie či typické správanie používateľov. Identifikované vzory slúžia na odhaľovanie úzkeho hrdla vo webovom sídle, predikciu správania veľkého množstva používateľov či odhaľovanie ich aktuálneho zámeru. Existujúce prístupy však využívajú najmä spoločné správanie množstva používateľov a globálne vzory správania.

V súčasnosti pozorujeme silnú tendenciu personalizovať Web a zameriavať sa na konkrétne potreby jednotlivcov. Cieľom práce je využiť tento trend aj pri vzoroch správania, kde je vhodné zachytiť správanie konkrétnych používateľov, sledovať ich odchýlky voči nájdeným globálnym vzorom a súvislosti so stereotypmi používateľov. Pre personalizáciu vzorov je vhodné správanie používateľa opísať z viacerých pohľadov - sledovať jeho akcie, vlastnosti navštíveného obsahu, či aktuálny kontext. Personalizované vzory je následne možné využiť na predikciu správania konkrétneho používateľa, odporúčanie či personalizáciu webového sídla.

Analyzujte existujúce riešenia reprezentácie a aplikácie typických vzorov správania používateľov webového sídla. Navrhnite prístup reprezentácie vzorov správania používateľa webového sídla, ktorý bude spájať znalosti získané z rôznych informačných zdrojov s dôrazom na individuálne charakteristiky správania používateľov doplnené globálnymi vzormi. Experimentálne overte navrhnutý prístup na netriviálnej vzorke dát.

---

[1] Vytlačiť obojstranne na jeden list papiera

[2] 150-200 slov (1200-1700 znakov), ktoré opisujú výskumný problém v kontexte súčasného stavu vrátane motivácie a smerov riešenia

Literatúra[3]

- Haibin Liu and Vlado Kešelj. 2007. Combined mining of Web server logs and web contents for classifying user navigation patterns and predicting users' future requests. Data Knowl. Eng. 61, 2 (May 2007), 304-330. DOI=http://dx.doi.org/10.1016/j.datak.2006.06.001
- Mehrdad Jalali, Norwati Mustapha, Md. Nasir Sulaiman, and Ali Mamat. 2010. WebPUM: A Web-based recommendation system to predict user future movements. Expert Syst. Appl. 37, 9 (September 2010), 6201-6212. DOI=http://dx.doi.org/10.1016/j.eswa.2010.02.105

Vyššie je uvedený návrh diplomového projektu, ktorý vypracoval(a) Bc. Tomáš Chovaňák, konzultoval(a) a osvojil(a) si ho Ing. Ondrej Kaššák a súhlasí, že bude takýto projekt viesť v prípade, že bude pridelený tomuto študentovi.

V Bratislave dňa 19.1.2016

Podpis študenta

Podpis výskumníka

Vyjadrenie garanta predmetov Diplomový projekt I, II, III

Návrh zadania schválený: áno / ~~nie~~[4]

Dňa: 15. 2. 2016

Podpis garanta predmetov

[3] 2 vedecké zdroje, každý v samostatnej rubrike a s údajmi zodpovedajúcimi bibliografickým odkazom podľa normy STN ISO 690, ktoré sa viažu k téme zadania a preukazujú výskumnú povahu problému a jeho aktuálnosť (uveďte všetky potrebné údaje na identifikáciu zdroja, pričom uprednostnite vedecké príspevky v časopisoch a medzinárodných konferenciách)

[4] Nehodiace sa prečiarknite

Čestne vyhlasujem, že som túto prácu vypracoval samostatne, na základe konzultácií a s použitím uvedenej literatúry.

V Bratislave, 11.5.2017

Tomáš Chovaňák

Anotácia

Slovenská technická univerzita v Bratislave

FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ

Študijný program: Informačné systémy

Autor: Bc. Tomáš Chovaňák

Diplomová práca: Rozpoznávanie vzorov správania používateľov webového sídla

Vedúci diplomovej práce: Ing. Ondrej Kaššák

máj 2017

Typické a opakujúce sa črty správania používateľov počas návštev webového sídla môžu byť vyjadrené pomocou vzorov správania reprezentovaných ako napr. frekventované množiny či sekvencie akcií, asociačné pravidlá, zhluky podobných sedení používateľov. Takéto vzory môžu byť využité napr. na identifikáciu preferencií používateľa, odporúčanie zaujímavého obsahu, predikciu ďalších akcií.

Existujúce metódy identifikujú vzory správania presnými ale výpočtovo a časovo náročnými spôsobmi. Vzory správania v týchto metódach sú príliš všeobecné a nevystihujú správanie špecifických skupín používateľov. Preto navrhujeme metódu, ktorá hľadá a kombinuje globálne vzory správania všetkých používateľov so vzormi správania dynamicky identifikovaných skupín podobných používateľov webového sídla. Používa algoritmy dolovania z prúdov dát, ktoré dokážu dáta spracovať rýchlo, jednoprechodovo a adaptovať sa na zmeny v nich. Nájdené vzory správania reprezentované ako uzavreté frekventované množiny akcií používame na odporúčanie zaujímavého obsahu používateľom podľa ich aktuálneho správania.

Metóda bola overená na dátach z domény e-learningu a spravodajstva. Môže byť použitá v dynamicky meniacom sa webovom sídle, kde rýchlo pribúdajú nové dáta a správanie používateľov sa často mení.

Annotation

Slovak University of Technology Bratislava

FACULTY OF INFORMATICS AND INFORMATION TECHNOLOGIES

Degree Course: Information systems

Author: Bc. Tomáš Chovaňák

Master's Thesis: Recognition of website users' behavioral patterns

Supervisor: Ing. Ondrej Kaššák

2017, May

Typical and repeating features of users' behavior during their visits of the website can be expressed through behavioral patterns represented e.g., as frequent itemsets or sequences of actions, association rules, clusters of similar user sessions. Such patterns can be used to identify user preferences, recommend interesting content, predict next actions etc.

Existing methods identify patterns in accurate but time and computationally expensive ways. Moreover, these patterns are too broad and general to describe behaviour of specific user groups well. Therefore, we propose method of behavioral patterns mining, which combines global patterns with patterns specific for dynamically identified groups of similar users. Our method makes use of well-known data stream algorithms to process data in online time by single pass and quickly adapt to changes. We represent behavioral patterns as frequent closed itemsets of user actions and use them for recommending interesting content to user according his actual behavior.

The method was evaluated on data from e-learning and news domains. The proposed method can be used in dynamically changing websites, with fast increase of new data and frequent changes in users' behavior.

Obsah

1 Úvod

Tak ako to dobrí obchodníci iste vedia, pochopenie zákazníka je nesmierne dôležité. Webové sídla sú dnes okrem iného vo veľkej miere aj miestom obchodu a reklamy. Aj tu sa ako kedysi na klasickom trhu obchodníci zjednávajú na cene a ponuke produktov so svojimi zákazníkmi, len to robia často nepriamo a to najmä prostredníctvom analýzy nazbieraných dát. Samozrejme nie vždy ide iba o obchod. Odhalenie zámerov návštevníka webového sídla a poznanie jeho správania slúži i jemu samému. Ide o vzájomný profit producentov a konzumentov služieb webového sídla. Napríklad ak ide o elektronický obchod producent chce zarobiť predajom a konzument zas získať správny produkt. Alebo spravodajský portál, kde producent chce získať čo najvyššiu čítanosť a konzument chce získať informácie.

Častým zdrojom informácií o správaní používateľov vo webovom sídle sú webové logy a v nich zachytené akcie, ktoré vykonali používatelia počas návštevy webového sídla, spájané do používateľských sedení. Tieto akcie sú implicitnou spätnou väzbou a teda objektívnym zdrojom informácií o správaní používateľov.

Správanie človeka je unikátne a nie je deterministické, podlieha aktuálnemu cieľu, kontextu či predchádzajúcej znalosti. Pri dlhodobejšom sledovaní používateľa však možno sledovať opakujúce sa črty správania ako napr. množiny a sekvencie spoločne vykonávaných akcií. Často opakujúce sa črty správania jedného či viacerých používateľov môžeme považovať za vzory správania. Podľa toho ako často sa vyskytujú im možno priradiť rôznu váhu. Niektoré vzory sú relevantné pre používateľa iba krátku dobu v závislosti od výnimočného kontextu (napr. čítanie športových správ počas olympiády), niektoré sú dlhodobejšie (napr. zvyk vždy ráno čítať správy zo sveta). Vzory správania môžu byť spoločné pre všetkých používateľov webového sídla, ale tiež pre menšie skupiny používateľov so špecifickým správaním, či len pre jednotlivcov. Identifikácia aktuálnych vzorov správania jednotlivcov, skupín a celej komunity môže viesť k posunu v chápaní používateľov, k spresňovaniu výsledkov v úlohách predikcie a odporúčania, či k vhodným biznis rozhodnutiam (napr. o segmentácii trhu).

Vo svete veľkých webových sídel s veľkým počtom používateľov je neustále generované množstvo dát o správaní používateľov. Preto je vhodné tieto dáta spracovávať jednoprechodovo ako potencionálne nekonečný prúd. Pribúdanie nových dát do webového sídla (napr. v spravodajskom portáli pribúdajúce články) vedie k zmenám v správaní používateľov, na ktoré je vhodné čo najrýchlejšie reagovať.

V tejto práci navrhujeme spôsob objavovania čo najaktuálnejších a relevantných vzorov správania, ktoré nehovoria len o správaní globálnej komunity používateľov, ale aj správaní menších špecifických subkomunít a jednotlivcov k nim patriacich. Navrhujeme tiež spôsob aplikácie nájdených vzorov v úlohe odporúčania zaujímavých položiek používateľom.

V prvej časti práce sa venujeme analýze problematiky dolovania z dát na Webe a vo webovom sídle. V kapitole 2 vysvetľujeme samotný pojem dolovania z dát na Webe ako

súčasť širšieho výskumu a potom opisujeme aj jednotlivé kroky procesu dolovania z dát na Webe a tiež možné aplikácie nájdených znalostí.

Následne v kapitole 3 opisujeme pojem vzor správania a jeho možné reprezentácie, približujeme niektoré konkrétne spôsoby dolovania vzorov správania v statických dátach reprezentovaných ako frekventované množiny a frekventované sekvencie. Tiež uvádzame príklady na niektoré ďalšie možné reprezentácie vzorov správania a metódy ich hľadania a aplikácie.

V kapitole 4 sa venujeme spôsobom dolovania vzorov správania z prúdu dát. Opisujeme špecifické požiadavky na algoritmy, ktoré spracovávajú prúd dát. Približujeme charakteristiky známych algoritmov dolovania frekventovaných množín z prúdu dát.

V kapitole 5 špecifikujeme ciele práce z ktorých vychádza návrh metódy v kapitole 6. Jej hlavným cieľom je hľadať nielen vzory správania spoločne pre celú komunitu používateľov, ale zároveň dynamicky segmentovať používateľov do skupín podľa ich podobného správania a získavať vzory typické pre identifikované skupiny. Špecifikom navrhovanej metódy je, že reaguje na výzvy súčasného trendu, kedy je potrebné spracovávať neustále a rýchlo generované veľké množstva dát. Dáta spracováva ako prúd a spĺňa požiadavky, ktoré sú na takýto typ spracovania dát kladené. V kapitole 7 opisujeme realizáciu nášho riešenia v podobe experimentálneho prototypu a prototypu pre distribuované prostredie. V kapitole 8 formulujeme hypotézy, opisujeme metodológiu a realizáciu experimentov v ktorých overujeme najmä prínos metódy v aplikácií na odporúčanie. Na záver v kapitole 9 zhŕňame podstatné poznatky, ktoré sme v tejto práci získali a opisujeme možný ďalší smer práce.

2 Získavanie znalostí z Webu

Dolovanie z dát na Webe (ang. *Web Mining*) predstavuje získavanie informácií z rozličných aspektov interakcie človeka na Webe. Výskum problematiky dolovania z dát na Webe je úzko prepojený aj s ďalšími výskumnými oblasťami ako napr. vyhľadávanie a extrakcia informácií, strojové učenie. Pri interakcií s Webom môžu byť techniky dolovania z dát použité na riešenia rôznych problémov (Kosala, 2000):

- hľadanie relevantných informácií (špecifickej informácie),
- získavanie nových znalostí z dostupných informácií,
- personalizácia informácií (obsahu a spôsobu prezentácie),
- spoznávanie záujmov používateľa (napr. pre účely marketingu).

Okrem toho tiež v práci (Kosala, 2000) rozdeľujú úlohu dolovania z dát na Webe do štyroch podúloh:

- Hľadanie webových zdrojov. Teda samotných webových dokumentov a dát.
- Selekcia špecifických informácií v predspracovaní týchto zdrojov.
- Zovšeobecňovanie. Teda automatizované hľadanie vzorov v individuálnych stránkach, alebo množine prepojených stránok.
- Analýza, validácia a interpretácia nájdených vzorov.

Možno rozlišovať tri kategórie dolovania z dát na Webe podľa časti webových zdrojov v ktorej sa uskutočňuje (Sisodia & Verma, 2012) :

- Dolovanie z obsahu Webu (ang. *Web content mining*). Predstavuje dolovanie informácií zo samotného obsahu webových dokumentov (teda vnútornej štruktúry dokumentov).
- Dolovanie zo štruktúry Webu (ang. *Web structure mining*). Predstavuje hľadanie modelu štruktúry prepojení medzi stránkami a zvlášť hľadanie zaujímavých vzťahov v tomto modeli.
- Dolovanie z dát o používaní Webu (ang. *Web usage mining*, skr. WUM). Predstavuje dolovanie z dát generovaných správaním používateľov Webu. Teda dát, ktoré zaznamenávajú interakciu používateľa s Webom. Hľadanie vzorov správania používateľov je tiež úlohou vyplývajúcou práve z tejto kategórie.

WUM je zaujímavou aplikáciou techník dolovania z dát na získavanie znalostí o správaní používateľov na Webe. Proces aplikácie WUM na personalizáciu webového sídla, ktorý nás zvlášť zaujíma, je zvyčajne zložený z dvoch fáz a to offline fázy, ktorá sa zaoberá získavaním a prípravou dát, trénovaním modelov a online fázy aplikovania nájdených znalostí. Jednou z výhod použitia personalizácie webového sídla založenej na znalostiach získaných pomocou WUM je, že vstupom nie sú subjektívne hodnotenia používateľov (explicitná spätná väzba), ale objektívne skutočnosti ako sú akcie používateľa v systéme (implicitná spätná väzba) (Mobasher, Dai, Luo, Sun, & Zhu, 2000). Tento proces pozostáva zo štyroch krokov (Sisodia & Verma, 2012):

1. Zbieranie dát z webových zdrojov,
2. príprava a predspracovanie dát,

3. hľadanie vzorov v dátach,
4. analýza, validácia nájdených vzorov a ich použitie v rôznych aplikáciách.

V časti 2.1 sa krátko venujeme možnostiam získavania a predspracovania dát z webových zdrojov. V časti 2.2 opisujeme možné aplikácie nájdených znalostí o správaní používateľov. Rôznym spôsobom reprezentácie a hľadania vzorov správania používateľov sa venujeme podrobnejšie aj v ďalších kapitolách 3 a 4.

2.1 Získavanie a predspracovanie dát

Prvým krokom spomínaného procesu WUM je získavanie dát o správaní používateľov. Zdrojmi týchto dát sú najmä logy webových serverov, proxy serverov a prehliadačov, používateľské profily a dáta získané pri registrácií používateľov vo webovom sídle, sedenia používateľov, cookies, dopyty používateľov, kliknutia a akékoľvek iné dáta, ktoré sú výsledkom interakcie používateľa s Webom (Sisodia & Verma, 2012).

Veľmi častým a dobre dostupným zdrojom dát o správaní používateľov sú logy webových serverov. Konzorcium W3C definuje štandardný odporúčaný formát webových logov[1], ale samozrejme existujú aj proprietárne formáty. Použitie štandardného formátu umožňuje využiť niektoré existujúce analytické nástroje, ktoré dokážu tento formát spracovávať[2]. Nazbierané dáta zaznamenané vo webových logoch typicky zdieľajú najmä tieto informácie: dátum, čas, IP klienta, autorizácia klienta, IP servera, port servera, metóda požiadavky, HTTP stavový kód, referenčná adresa, host, používateľský agent, čas vykonania akcie, počet prenesených bajtov a ďalšie.

Webové logy sú typicky veľmi veľké textové súbory (rádovo gigabajty dát). Záznamy v nich sú usporiadané chronologicky. Webové logy môžeme rozdeliť podľa typu informácií, ktoré zachytávajú. Zo špecifickejších logov vieme nachádzať špecifickejšie vzory (Sisodia & Verma, 2012):

- Prístupové logy – zachytávajú detailné informácie o požiadavkách posielaných z prehliadača na webový server.
- Chybové logy – zachytávajú informácie o akýchkoľvek chybách a požiadavkách na webový server, ktoré zlyhali.
- Logy prehliadačov – zachytávajú informácie o prehliadači a operačnom systéme používateľov, ktorý posiela požiadavky na webový server.
- Logy odkazujúcich serverov – zachytávajú URL stránok, ktoré sa odkazujú na skúmanú stránku a umožňujú tak zistiť odkiaľ používatelia do webového sídla prichádzajú.

Najmä kvôli obmedzeniam daným špecifikáciou protokolu HTTP sú údaje v bežných webových logov nekompletné a neumožňujú napr. jednoznačnú identifikáciu používateľa

[1] http://www.w3.org/TR/WD-logfile.html

[2] napr. https://www.weblogexpert.com/info/IISLogs.htm

a teda ani jeho používateľských sedení. Vo svojej práci Spiliopoulou a kol. (Spiliopoulou, 2003) klasifikujú prístupy k získavaniu dát o používaní Webu do dvoch skupín:

- reaktívne prístupy,
- proaktívne prístupy.

Reaktívny prístup používa existujúce webové logy, ktoré vznikli všeobecne a pravdepodobne bez zámeru použitia pre aplikácie WUM na personalizáciu a teda tiež bez identifikovania používateľov a ich sedení. Reaktívny prístup sa snaží transformovať existujúce webové logy, identifikovať používateľov a ich sedenia. Kvôli tomu, že tento prístup nevyžaduje žiaden špecializovaný softvér, ale využíva len všeobecné logy je pomerne rozšírený. Jeho problémom je však, že logy nie sú prispôsobené na takýto typ úlohy (Huntington, Nicholas, & Jamali, 2008).

Proaktívny prístup zbiera dáta spôsobom, ktorý je navrhnutý so zámerom pre konkrétnu aplikáciu. Na najdôležitejšie úlohy z pohľadu prípravy dát v rámci WUM, teda identifikáciu používateľov a ich sedení, využíva explicitné mechanizmy.

Surové webové logy často obsahujú množstvo irelevantných častí. Dôležitým predpokladom pre zabezpečenie správnosti znalostí získaných pomocou WUM je teda správna príprava dát. V rámci predspracovania dát je často potrebné riešiť problémy v krátkosti predstavené v nasledujúcich častiach.

Očistenie dát

Očistenie dát znamená zbavenie sa nepotrebných záznamov ako napr. neúspešných požiadaviek, či požiadaviek generovaných webovými robotmi, požiadaviek s metódami inými ako GET a POST, suplementárnych požiadaviek napr. na obrazový obsah stránok alebo rôznych dodatočných skriptov (Srivastava, Garg, & Mishra, 2014).

Identifikácia používateľov

Pri väčšine webových serverov sú používatelia anonymní a nevieme presne určiť priradenie logovaných akcií k používateľom. Pri reaktívnom prístupe je jednou zo stratégií identifikácie používateľov aproximácia ich identity z IP adries, operačného systému zariadenia z ktorého sa pripája, prehliadača, ktorý používa a ďalších informácií. Táto stratégia však samozrejme nedokáže presnejšie identifikovať konkrétneho používateľa napr. v prípadoch, keď jedno zariadenie využívajú viacerí používatelia, alebo jeden používateľ používa viacero zariadení. Jednou z ďalších možností spresnenia implicitného identifikovania používateľa, je napr. použitie behaviorálnych profilov. V tomto prípade však treba získať trénovacie dáta, ktoré majú explicitne uvedené id používateľa. (Srivastava, Garg, & Mishra, 2014)

Pri proaktívnom prístupe je ďalšou možnosťou identifikácie používateľov sprístupnenie obsahu webového sídla až po autentifikácií používateľa, čo je však mnohokrát odradzujúcim prvkom. Ak autentifikácia nie je cesta, je možné použiť cookies. Problém s cookies je, že používatelia môžu mať v prehliadači vypnuté ich používanie alebo ich môžu kedykoľvek vymazať. (Huntington, Nicholas, & Jamali, 2008)

Identifikácia používateľských sedení

Používateľské sedenie predstavuje množinu akcií, ktoré používateľ vykonal v definovanom časovom rozsahu. Jedna z často používaných stratégií identifikácie používateľských sedení pri reaktívnom prístupe je založená na predpoklade, že ďalšie sedenie používateľa začína až po prekročení stanoveného časového limitu medzi dvoma vykonanými akciami (často sa využíva limit 30 min (Herder, 2007)). Používané časové limity by mali vychádzať z výstupov štatistických experimentov (Srivastava, Garg, & Mishra, 2014).

Pri proaktívnom prístupe môže mať webový server implementovaný vlastný mechanizmus identifikácie používateľských sedení (Huntington, Nicholas, & Jamali, 2008). Ten priradzuje unikátny identifikátor všetkým požiadavkám klienta až kým neexpiruje, alebo sa neukončí spojenie. Hlavným problémom je, že skutočný koniec sedenia aj napriek týmto mechanizmom ostáva neznámy.

Kompletizácia ciest

Niektoré cesty a prepojenia stránok v identifikovaných používateľských sedeniach môžu byť nekompletné, kvôli tomu, že niektoré ich prístupy nie sú zaznamenané v logoch (Srivastava, Garg, & Mishra, 2014). Môže to byť napr. kvôli použitiu tlačidla späť v prehliadači, alebo priamemu zadaniu URL. Preto je niekedy potrebné dodatočne nájsť existujúce a využité prepojenia medzi jednotlivými stránkami navštívenými v rámci používateľského sedenia.

2.2 Využitie a aplikácia dát o používaní webového sídla

Finálnym krokom procesu WUM je použitie a aplikácia získaných informácií o správaní používateľov, ktoré ďalej v práci označujeme aj ako vzory správania (v ďalších kapitolách sa bližšie venujeme spôsobom ich reprezentácií a hľadania). Všeobecným cieľom WUM je zbierať zaujímavé informácie o správaní používateľov a použiť ich na vylepšenie webového sídla z ich hľadiska (Facca & Lanzi, 2003). V tejto podkapitole opisujeme jednotlivé oblasti aplikácie informácií získaných v procese WUM podľa (Facca & Lanzi, 2003) a uvádzame konkrétne príklady.

Personalizácia webového sídla

Pomocou porovnávania aktuálneho správania používateľov s nájdenými vzormi je možné využiť tieto vzory na predikciu ďalších krokov používateľa, odporúčanie produktov a položiek, inú adaptáciu webových sídel podľa aktuálnych potrieb používateľa. Existuje viacero prác zaoberajúcich sa aplikáciou vzorov správania na personalizáciu, ktoré sa líšia najmä spôsobom ich hľadania a reprezentácie.

Metóda *WebPUM* (Jalali, Mustapha, Nasir Sulaiman, & Mamat, 2010) reprezentuje vzory správania ako komponenty získané po aplikácií algoritmu delenia grafu na graf reprezentujúci prepojenia medzi stránkami webového sídla (vytvorený z dát predspracovaného webového logu). Uzly grafu reprezentujú webové stránky a hrany grafu ich vzájomné váhované prepojenia. Váha prepojení je odvodená od frekvencie a intenzity návštev dvojíc stránok používateľmi v rovnakých sedeniach. Získané vzory správania

používajú na generovanie odporúčaní podľa aktuálneho správania používateľov. Hľadajú vzory najpodobnejšie posledným krokom používateľa. Odporúčajú položky, ktoré sa nachádzajú v týchto vzoroch a ktoré používateľ ešte nenavštívil.

Metóda na predikciu akcií používateľov je opísaná v práci (Liraki & Harounabadi, 2015). Používa fuzzy c-means zhlukovanie na hľadanie zhlukov podobných sedení používateľov a stupňa prepojení medzi používateľmi a nájdenými zhlukmi. Vzory správania sú reprezentované ako váhované asociačné pravidlá získané z každého nájdeného zhluku sedení. V práci vyhodnocujú úspešnosť použitia vzorov správania na predikciu ďalších krokov používateľa.

V práci (Anandhi & Irfan Ahmed, 2014) navrhujú použitie zhlukovacieho algoritmu DBSCAN na hľadanie vzorov v dátach o používaní Webu. DBSCAN je založený na hustote dátových bodov, čo zvyšuje pravdepodobnosť nájdenia vzorov s nízkou podporou (frekvenciou výskytov), ale vysokou spoľahlivosťou, ktoré by napr. pri použití algoritmu hľadania asociačných pravidiel s definovanou hranicou minimálnej podpory boli ignorované. Ich ďalším prínosom je návrh použitia invertovaného indexu pre efektívnejšiu online predikciu správania používateľa.

Ukladanie stránok do dočasnej pamäti a prednačítavanie

To, že sú vzory správania použiteľné v metódach predikcie správania používateľa je možné využiť aj na zvýšenie výkonnosti a času odozvy webového servera. Stránky predikované na základe aktuálneho správania používateľa, ktoré s vysokou pravdepodobnosťou v ďalších krokoch navštívi sa prednačítavajú a ukladajú do dočasnej pamäte z ktorej sú potom veľmi rýchlo získané.

V práci (Teng, Chang, & Chen, 2005) opisujú využitie vzorov správania v podobe asociačných pravidiel na predikciu správania používateľa. Navrhujú algoritmus, ktorý tieto pravidlá efektívne využíva na ukladanie predikovaných stránok do dočasnej pamäte, integrovaný s prednačítavaním stránok na strane klienta. V rámci tohto algoritmu navrhli tiež funkciu ohodnotenia profitu z prípadného prednačítania konkrétneho objektu uvažujúc mnohé aspekty (napr. veľkosť objektu, počet referencií na objekt, spoľahlivosť použitého asociačného pravidla), vďaka ktorej vedia určiť či je daný objekt vhodné prednačítať alebo nie.

Podpora pre zlepšovanie dizajnu webového sídla

Ako sme už uviedli, dáta o správaní používateľov z webových logov predstavujú implicitnú a teda objektívnu spätnú väzbu, ktorá je veľmi dobrým zdrojom aj pre účely zlepšovania použiteľnosti webového sídla. Získané znalosti o správaní používateľa môžu byť použité na adaptáciu štruktúry a dizajnu webového sídla tak aby sa zrýchlil napr. čas vyhľadania často navštevovaných stránok, položiek a pod. V práci (Perkowitz & Etzioni, 1997) definujú pojem adaptívneho webového sídla a výzvu na automatizovanie procesu reorganizácie štruktúry webového sídla podľa aktuálneho správania používateľov. Práca (Fu, Shih, Creado, & Ju, 2002) je príkladom použitia WUM na adaptáciu dizajnu a štruktúry podľa správania používateľov. Ich cieľom je reorganizovať štruktúru webového sídla tak aby sa dalo používateľom dostať k informáciám pomocou čo najmenšieho počtu

kliknutí. Stránky klasifikujú ako indexy a obsahové stránky podľa toho či obsahujú hlavne linky na ďalšie položky, resp. skôr textový obsah. Jedným z viacerých príkladov reorganizácie je, že často navštevované stránky klasifikované ako obsahové umiestňujú automaticky vyššie v hierarchickej štruktúre. Malými zmenami v štruktúre sa snažia dosiahnuť postupnú adaptáciu štruktúry webového sídla podľa potrieb používateľov.

Porozumenie správaniu používateľa pre podporu biznis rozhodnutí

Pohľad na správanie používateľov webového sídla cez vysokoúrovňové vzory správania môže slúžiť ako základ pre dôležité rozhodnutia v biznis pozadí webového sídla. Napríklad rozhodnutia o segmentácií trhu, pomocou ktorej sa prispôsobujú marketingové aktivity skupinám zákazníkov podľa ich spoločných záujmov. Existujú nástroje podporujúce takéto rozhodovanie od malých lacných nástrojov analyzujúcich webové logy (ako napr. WebLog Expert [3]) po zložitejšie riešenia integrované aj v rámci CRM systémov (napr. WebTrends[4]).

2.3 Sumarizácia

V tejto časti sme opísali, čo znamená dolovanie z dát na Webe, z akých krokov pozostáva a aké má praktické využitie. Z podkategórií dolovania z dát na Webe nás ďalej v tejto práci zaujíma dolovanie z dát o používaní Webu (ang. *Web usage mining* a skr. WUM). WUM je proces, ktorý je možné automatizovať a pozostáva z hlavných krokov: predspracovanie dát, hľadanie vzorov v dátach a ich následná aplikácia na rôzne účely.

Stručne sme predstavili niekoľko problémov týkajúcich sa predspracovania dát z webových logov. Týmito problémami sa nezaoberáme príliš do hĺbky, pretože sú sami osebe širokou problematikou. V tejto práci sa viac venujeme samotnému hľadaniu vzorov z predspracovaných dát o používaní webového sídla a ich aplikácií v úlohe odporúčania.

V aplikáciách, ktoré sme opísali, sú často oddelené kroky predspracovania a hľadania vzorov, ktoré sú vykonávané v rámci offline komponentu, od aplikácie nájdených vzorov vykonávanej v rámci online komponentu. Tu sa otvára výzva na hľadanie takej reprezentácie vzorov správania a metódy ich hľadania aby bolo možné všetky kroky WUM procesu vykonávať v jednom online komponente a dáta o používaní webového sídla spracovávať flexibilne jednoprechodovo ako potencionálne nekonečný prúd dát.

Výhodou vzorov nájdených pomocou WUM, oproti evaluácií webového sídla používateľmi prostredníctvom štúdie, testovania alebo dotazníka je nielen to, že ich hľadanie a analýza môžu byť automatizované, ale hlavne to, že nezáležia od dobrovoľného subjektívneho vstupu používateľov (explicitnej spätnej väzby), ale od objektívnych akcií používateľa vo webovom sídle (Anandhi & Irfan Ahmed, 2014). Sú teda vynikajúcim zdrojom znalostí pre analýzu správania používateľov.

[3] https://www.weblogexpert.com/

[4] https://www.webtrends.com

Samotná aplikácia nájdených vzorov môže byť rôzna, uviedli sme niekoľko prác, kde opisujú jednak rôzne metódy hľadania vzorov správania v rôznych reprezentáciách a tiež rôzne spôsoby ich využitia. Ďalšími možnými spôsobmi reprezentácie vzorov správania a presnejšou definíciou vybraných reprezentácií sa zaoberáme aj v ďalších kapitolách.

3 Vzory správania v statických datasetoch

Vzory správania nie sú pevne definovaným pojmom a vo webovom sídle si ich môžeme predstavovať ako rôzne skutočnosti. Napr. sekvencie často po sebe navštívených stránok, typické stránky ktorými začínajú sedenia používateľov, postupnosti navštevovaných sekcií webového sídla, priemerné dĺžky návštev stránok, priemerné dĺžky sedení, a mnoho ďalších identifikovaných pravidelne sa opakujúcich atribútov súvisiacich so správaním používateľa vo webovom sídle.

V tejto práci vzory správania používateľov webového sídla rozumieme najčastejšie ako v literatúre bežné uvádzané tzv. frekventované vzory (ang. *frequent pattern*) získané z dát zachytávajúcich správanie používateľov webového sídla. Ako sa uvádza v práci (Han, Kamber, & Pei, 2012) frekventovaný vzor zachytáva často opakujúce sa vzťahy v dátach. Frekventované vzory môžu mať rôznu podobu – všeobecne to môžu byť často spolu sa vyskytujúce neusporiadané podmnožiny z množiny všetkých položiek, usporiadané subsekvencie celých sekvencií položiek, či iné subštruktúry celkovej štruktúry. Frekventované vzory sú vstupom pre hľadanie asociačných pravidiel (ktoré môžu byť tiež chápané ako reprezentácia vzorov správania) a môžu byť nápomocné aj v ďalších úlohach dolovania z dát ako napr. klasifikácia, zhlukovanie.

Frekventované vzory môžeme podľa toho či zachovávajú sekvenčnosť položiek alebo nie rozdeliť na (Han, Kamber, & Pei, 2012):

- frekventované množiny (ang. *frequent itemsets*), ktoré ignorujú poradie položiek. Predstavujú teda často spolu vyskytujúce sa položky vo vstupných dátach. Spôsobu ich hľadania sa venujeme v časti 3.1.
- frekventované sekvencie (ang. *frequent sequences*), ktoré zachovávajú sekvenčnosť položiek v sekvenciách. Spôsobu ich hľadania sa venujeme v časti 3.2.

V časti 3.3 sa krátko venujeme aj iným prístupom reprezentácie vzorov správania používateľov a spôsobu ich získavania napr. zhlukovaním sedení používateľov, rozdeľovaním grafu reprezentujúceho správanie používateľov.

3.1 Dolovanie frekventovaných množín

Nech D je databáza transakcií. Nech $C = \{i_1, i_2, i_3, \dots, i_n\}$ je množina všetkých položiek v D. Položkou i môže byť napríklad záznam o navštívení stránky webového sídla používateľom zachytený vo webových logoch, alebo čokoľvek iné ako napr. položka v košíku používateľa v internetovom obchode. Transakcia môže predstavovať napríklad sedenie používateľa, či košík používateľa v internetovom obchode. Transakcia T je teda množinou položiek. Platí $T \subseteq C$. Frekventovaná množina je akákoľvek množina $A \subset T$ s hodnotou podpory (počet výskytov v dátach) vyššou ako vopred určená hranica minimálnej podpory (ang. *minimum support threshold*). Nech kardinalita frekventovanej množiny A je k. Takúto frekventovanú množinu budeme nazývať *k-množinou*. Podpora množiny je vypočítaná ako absolútna podpora, teda počet transakcií v ktorých sa vyskytuje, alebo ako relatívna podpora, teda pomer počtu výskytov množiny k počtu všetkých

transakcií v celej databáze (čo je vlastne pravdepodobnosť výskytu danej množiny v transakcií patriacej uvažovanej databáze).

Dôležitým problémom pri dolovaní frekventovaných množín ako sa uvádza aj v (Han, Kamber, & Pei, 2012) je, že sa generuje často obrovské množstvo množín, ktoré spĺňajú hranicu minimálnej podpory (zvlášť ak je táto nastavená príliš nízko). Je to dôsledok tzv. *apriori* vlastnosti, ktorá hovorí, že akákoľvek podmnožina frekventovanej množiny je tiež frekventovaná.

Ako riešenie tohto problému boli navrhnuté koncepty tzv. uzavretých frekventovaných množín a maximálnych frekventovaných množín ako sa uvádza v (Chi Y. a., 2006):

- Množina *A* je ***maximálna frekventovaná množina*** ak neexistuje množina *B* taká, že $A \subset B$ a zároveň *B* je frekventovaná množina. Nech *M* je množina všetkých maximálnych frekventovaných množín a *F* je množina všetkých frekventovaných množín. Platí, že *M* je oveľa menšie ako *F.* Tiež platí, že vieme z množiny *M* získať *F* ak doplníme do *M* všetky podmnožiny *M* (keďže na základe *apriori* znalosti platí, že všetky tieto podmnožiny budú tiež frekventované). Problémom tohto prístupu je, že dolovaním maximálnych frekventovaných množín sa stráca informácia o podpore jednotlivých dopočítavaných frekventovaných množín.
- Množina *A* je ***uzavretá množina*** ak neexistuje množina *B* taká, že $A \subset B$ a *B* má rovnakú hodnotu podpory ako *A*. Nech *C* je množina všetkých frekventovaných uzavretých množín. Nech *F* je množina všetkých frekventovaných množín. Aj pri *C* platí podobne ako pri maximálnych frekventovaných množinách, že veľkosť *C* je stále výrazne menšia ako *F.* Presnejšie platí: $M \subseteq C \subseteq F$ .Tiež platí, že vieme zostrojiť množinu *F* z množiny *C,* keďže každá frekventovaná množina je buď uzavretá, alebo je podmnožinou jednej alebo viacerých frekventovaných uzavretých množín. Takisto vieme dopočítať podporu všetkých frekventovaných množín. Vieme, že podpora frekventovanej množiny bude rovnaká ako maximálna z podpôr uzavretých frekventovaných množín, ktoré ju obsahujú.

Na dolovanie frekventovaných množín bolo od doby kedy bola takáto úloha definovaná navrhnutých viacero algoritmov. Algoritmy hľadania frekventovaných množín v statických datasetoch môžeme klasifikovať do niekoľkých základných skupín podľa toho na akom princípe fungujú (Aggarwal, Bhuiyan, & Hasan, 2014):

- **Algoritmy založené na spájaní množín.** Tieto algoritmy generujú kandidátov na frekventované *(k+1)-množiny* spájaním frekventovaných *k-množín*. Na zmenšenie prehľadávaného priestoru využívajú tzv. apriori princíp, ktorý hovorí, že všetky podmnožiny frekventovaných množín sú tiež frekventované množiny. Základným algoritmom tejto skupiny je algoritmus *Apriori* (Agrawal & Srikant, 1994). Aj napriek využitiu apriori princípu má však vážne problémy s výpočtovou náročnosťou spojenou s veľkým počtom generovaných kandidátnych množín. Existuje viacero algoritmov, ktoré sa snažia o jeho optimalizáciu. Algoritmus Apriori a jeho varianty opisujeme podrobnejšie v prílohe A.
- **Algoritmy založené na stromovej štruktúre.** Kandidáti na frekventované množiny môžu byť objavovaní pomocou stromovej štruktúry tzv. lexikografického

stromu. Koreň stromu je prázdny a uzly stromu reprezentujú lexikograficky usporiadané množiny. Nech $I = \{i_1, \dots i_k\}$ je množina korešpondujúca s uzlom stromu. Potom rodičom uzla je množina $\{i_1, \dots i_{k-1}\}$. Napríklad dve množiny *acdfh* a *acdfg* sú susedné uzly pretože sú potomkovia uzla *acdf*. Ich spojením vznikne kandidátna množina *acdfgh*. V podstate všetky algoritmy založené na apriori princípe možno považovať za algoritmy využívajúce lexikografický strom. Často sa líšia spôsobom konštrukcie tohto stromu. Algoritmus *Apriori* využíva stratégiu prehľadávania do šírky a lexikografický strom možno uvažovať len implicitne. Algoritmus *TreeProjection* (Agarwal, Aggarwal, & Prasad, 2001), existuje vo verzií so stratégiou prehľadávania do šírky aj do hĺbky a explicitne využíva lexikografický strom pre generovanie frekventovaných množín. Niektoré ďalšie algoritmy takisto existujú v modifikáciách podľa toho akú stratégiu prehľadávania stromu používajú (napr. *Eclat* (opisujeme ho v prílohe A) a jeho modifikácia *dEclat* (Zaki & Gouda, 2003). Jednou z výhod prístupu prehľadávania lexikografického stromu do hĺbky je rýchle nachádzanie dlhých maximálnych frekventovaných množín. Explicitné použitie lexikografického stromu je nápomocné tiež pri vizualizovaní stratégií prehľadávania kandidátnych frekventovaných množín a pochopeniu prínosu rôznych algoritmov. Tieto algoritmy často využívajú na zmenšenie priestoru potrebného pre počítanie podpory množín projekciu (napr. *TreeProjection, Eclat, dEclat)*, kedy na výpočet podpory s príponou *P* stačí prechádzať časť databázy transakcií obsahujúcu množinu *P*.

- **Algoritmy využívajúce vertikálny formát dát**. Nech *tid* označuje identifikátor transakcie, *tidlist* označuje množinu identifikátorov transakcií a *item* označuje položku. Základnou myšlienkou je zefektívnenie počítania podpory pomocou transformácie z horizontálneho formátu dát: <*tid, item*> na vertikálny formát dát <*item, tidlist*>, kde *tidlist* je zoznam transakcií v ktorých sa nachádza daná položka. Kľúčový princíp je, že je možné vypočítať podporu *k-množiny* vypočítaním prieniku dvoch množín transakcií *tidlist* spájaných (*k-1)-množín.* Príkladom takéhoto algoritmu je *Partition* (Savasere, 1995) a jeho neskorší nasledovník *Eclat* (Zaki M. J., 2000) (podrobnejšie v prílohe A).
- **Algoritmy založené rekurzívnom raste prípon vzorov.** Na rozdiel od väčšiny ostatných prístupov, kde sa hľadajú vzory v strome budovanom postupne z predpôn jednotlivých lexikograficky usporiadaných množín, je tento prístup založený na postupnom rozširovaní prípon frekventovaných množín. Prehľadávanie v prístupe s rastom prípon vzorov nie je principiálne odlišné od klasického prehľadávania s rastom predpôn vzorov (napríklad *TreeProjection* algoritmus) s rozdielom, že položky sú usporiadané od tej s najvyššou podporou po najnižšiu. Tieto prístupy nevyžadujú výpočtovo náročné generovanie kandidátnych sekvencií, čo je typický problém algoritmov založených na apriori princípe. Veľmi dôležitým algoritmom patriacim do tejto skupiny je *FP-Growth* (Han, Pei, & Yin, 2000). Jeho základom je vytvorenie vysoko kompaktnej štruktúry s názvom *FP-tree*, reprezentujúcej pôvodné transakčné dáta. Transakčná databáza je rekurzívne rozdeľovaná na menšie projektované časti podľa aktuálnych vzorov (prípon). Vzory postupne rastú o lokálne frekventované množiny hľadané v týchto častiach. Podrobne tento algoritmus opisujeme v prílohe A.

- **Algoritmy hľadajúce maximálne a uzavreté frekventované množiny.** Veľké množstvo nachádzaných frekventovaných množín býva informačne redundantných. Ako sme už uviedli, maximálne a uzavreté frekventované množiny sú kompaktnou reprezentáciou väčšieho množstva frekventovaných množín. Algoritmy z tejto skupiny sa zameriavajú práve na hľadanie týchto kompaktných reprezentácií, čo značne zmenšuje priestor prehľadávania. Príkladom algoritmu na dolovanie maximálnych frekventovaných množín sú *MaxMiner* (Bayardo & Roberto, 1998), *DepthProject* (Agarwal, Aggarwal, & Prasad, 2000), *MAFIA* (Burdick, Calimlim, & Gehrke, 2001). Príkladom algoritmov na dolovanie uzavretých frekventovaných množín je *CLOSET* (Pei, Han, & Mao, 2000) a *CHARM* (Zaki & Hsiao, 2002) (podrobnejšie v Prílohe A).

3.2 Dolovanie frekventovaných sekvencií

Úloha dolovania frekventovaných sekvencií bola prvýkrát definovaná v práci (Agrawal & Srikant, 1995). Dôležitým predpokladom je, že spracovávané dáta pozostávajú zo záznamov, ktoré sú sekvenčného charakteru. Príkladom môžu byť po sebe nasledujúce transakcie vykonané zákazníkmi elektronického obchodu či návštevy stránok webového sídla. Uvádzame definíciu problému dolovania frekventovaných sekvencií z (Agrawal & Srikant, 1995).

Nech *C* je kolekcia všetkých položiek $i \in C$ v dátach. Položkou i môže byť napríklad záznam o navštívení stránky webového sídla používateľom zachytený vo webových logoch. Nech *I* je neprázdnou množinou položiek, ktoré sa vyskytujú v dátach spoločne (pri sebe v sekvencií). Ak daná množina *I* obsahuje napríklad položky $a \in C$ a $b \in C$ tak ju môžeme zapísať aj takto: $I = (ab)$. Teda množinu položiek môžeme zapísať ako $I = (i_1, i_2, i_3, \dots, i_m)$. Sekvencia S je zoradeným zoznamom takýchto množín položiek. Teda $S = < s_1, s_2, s_3, \dots, s_n >$ kde s_i označuje v poradí i-tu množinu položiek. $S' = < s'_1, s'_2, s'_3, \dots, s'_n >$ je subsekvenciou $S = < s_1, s_2, s_3, \dots, s_m >$ ak existujú také indexy $1 \leq i_1 < i_2 < i_3 < \dots < i_n \leq m$, že $s'_1 \subseteq s_{i_1}, s'_2 \subseteq s_{i_2}, s'_3 \subseteq s_{i_3}, \dots, s'_n \subseteq s_{i_n}$. Nech *D* je databáza sekvencií. Úlohou je nájsť všetky frekventované sekvencie nachádzajúce sa v databáze *D*. Výpočet podpory frekventovanej sekvencie je vlastne výpočtom frekvencie výskytov danej sekvencie v *D*. Sekvencia je považovaná za frekventovanú (a teda možno ju označiť ako tzv. *sekvenčný vzor*) ak má podporu presahujúcu vopred určenú hranicu minimálnej podpory (ang. *minimum support threshold*).

Sekvenčný vzor s dĺžkou *l* sa nazýva *l-sekvenčný vzor.* Množina frekventovaných *l-sekvencií* nech je F_l. Ak neexistuje žiadna nadsekvencia sekvenčného vzoru $\alpha \in D$ s rovnakou podporou ako *α*, tak *α* je *uzavretý sekvenčný vzor* v databáze *D*. Sekvenčný vzor *α* je *maximálny sekvenčný vzor* ak ho neobsahuje žiadny iný sekvenčný vzor v databáze *D* (Shen, Wang, & Han, 2014).

Tabuľka 1 je ilustráciou možnej databázy sekvencií *D*. Nech je hodnota minimálnej podpory 2. Keďže sekvencie 1 a 3 obsahujú subsekvenciu $s = < (ab)c >$ tak *s* má hodnotu podpory 2 a je teda sekvenčným vzorom s dĺžkou 3. Sekvencia $\beta = < (ab)dc >$

je zas príkladom uzavretého sekvenčného vzoru keďže neexistuje taká nadsekvencia z *D*, ktorá by mala rovnakú hodnotu podpory (Shen, Wang, & Han, 2014).

Tabuľka 1. Príklad databázy sekvencií z (Shen, Wang, & Han, 2014).

ID	Sekvencia
1	<a(abc)(ac)d(cf)>
2	<(ad)c(bc)(ae)>
3	<(ef)(ab)(df)cb>
4	<eg(af)cbc>

Na dolovanie frekventovaných sekvencií bolo od doby kedy bola takáto úloha definovaná navrhnutých viacero algoritmov. Algoritmy hľadania frekventovaných sekvencií v statických datasetoch možno rozdeliť do základných skupín podľa toho na akom princípe fungujú (Shen, Wang, & Han, 2014):

- **Algoritmy založené na apriori princípe**. Apriori princíp je využitý na zmenšenie prehľadávaného priestoru. Hovorí, že všetky podmnožiny frekventovaných sekvencií sú tiež frekventované sekvencie. Ďalej možno algoritmy rozdeliť podľa toho aký formát dát využívajú:
  - **využívajúce horizontálny formát dát** (napr. Tabuľka 1). Patrí tu napr. algoritmus *AprioriAll,* ktorý bol navrhnutý v práci (Agrawal & Srikant, 1995) ako jedno z prvých riešení problému hľadania frekventovaných sekvencií. Ten je vlastne rozšírením techník používaných v známom algoritme *Apriori* určenom na hľadanie frekventovaných množín a asociačných pravidiel. Jedno z jeho vylepšení je algoritmus *GSP* (Srikant & Agrawal, 1996), ktorý generalizuje definíciu sekvenčných vzorov do ktorej zahŕňa možnosť využitia taxonómie položiek a rôznych časových obmedzení určujúcich existenciu sekvencií (podrobnejšie v Prílohe B).
  - **využívajúce vertikálny formát dát**. Predstavuje zoznam jednotlivých položiek *item* so zoznamom *slist* obsahujúcim referencie na sekvencie a množiny položiek v ktorých sa položky nachádzajú: <*item, slist*>. Tento formát zjednodušuje najmä počítanie podpory sekvencií pomocou prienikov zoznamov sekvencií a množín. Patrí tu napr. algoritmus *SPADE* (Zaki M. J., 2001), ktorý dokáže nájsť všetky frekventované sekvencie pomocou troch prechodov cez dáta a rozdeľuje priestor prehľadávania na nezávisle časti podľa prefixu sekvencií (podrobnejšie v Prílohe B).
- **Algoritmy založené na rekurzívnom raste vzorov**. Pomocou známeho prístupu rozdeľuj a panuj je sekvenčná databáza rekurzívne rozdeľovaná na menšie projektované časti, v ktorých sa sekvenčné vzory hľadajú oddelene. Vzory postupne rastú o lokálne frekventované sekvencie hľadané v týchto častiach. Tieto prístupy nevyžadujú výpočtovo náročné generovanie kandidátnych sekvencií, čo je typický problém algoritmov založených na apriori princípe. Patria tu napr. algoritmy *FreeSpan* (Han, et al., 2000) (podrobnejšie v prílohe B) a *PrefixSpan* (Pei J. , et al., 2001).

3.2.1 Rozširujúce prístupy k dolovaniu frekventovaných sekvencií

V tejto časti v krátkosti opisujeme niekoľko ďalších algoritmov a prístupov, ktoré rozširujú základné prístupy k dolovaniu frekventovaných sekvencií a snažia sa vyriešiť niektoré problémy s ktorými sa v praxi stýkajú.

Algoritmy hľadajúce uzavreté frekventované sekvencie

Podobne ako v oblasti dolovania frekventovaných množín sú to algoritmy ako *CHARM, CLOSET*, tak v oblasti dolovania frekventovaných sekvencií boli navrhnuté napr. algoritmy *CloSpan* (Yan, Han, & Afshar, 2003) a *BIDE* (Wang & Han, 2004) . Dolovanie uzavretých frekventovaných sekvencií výrazne zmenšuje oblasť prehľadávania a zvyšuje výkonnosť algoritmu najmä pri nízko nastavenej hladine minimálnej podpory a dolovaní dlhých frekventovaných sekvencií (Shen, Wang, & Han, 2014).

Algoritmy hľadajúce viacdimenzionálne sekvencie

Štandardné algoritmy hľadajú frekventované sekvencie v jedno alebo dvoj dimenzionálnom priestore. V reálnych situáciách bývajú sekvencie asociované s rôznymi atribútmi, ktoré formujú multidimenzionálny priestor. Napr. sekvencie nákupov zákazníkov sú asociované napr. s regiónom, časom, osobitnou zákazníckou skupinou a iné. Príkladom je algoritmus *Uni-Seq* navrhnutý v práci (Pinto, et al., 2001), kde multidimenzionálnu informáciu vkladajú ako ďalšiu množinu položiek do sekvencie. Touto transformáciou sa stane databáza sekvencií klasickou jednodimenzionálnou sekvenčnou databázou na ktorú v tomto prípade aplikujú algoritmus *PrefixSpan*. Výsledkom môžu byť zaujímavé znalosti ako napr. zistenie, že jedna skupina zákazníkov má celkom odlišné správanie ako iná (Shen, Wang, & Han, 2014).

Aproximatívne metódy

Bežné metódy hľadania sekvenčných vzorov majú problémy s databázami obsahujúcimi mnoho dlhých sekvencií a šumu, kvôli čomu generujú množstvo krátkych a triviálnych vzorov, ale nedokážu nájsť skutočne zaujímavé vzory. V práci (Kum, Pei, Wang, & Duncan, 2003) navrhli pojem *dolovanie aproximatívnych sekvenčných vzorov.* Ide o to, že namiesto hľadania exaktných vzorov sa hľadajú vzory zdieľané mnohými sekvenciami v databáze. Nimi navrhnutý algoritmus *ApproxMAP* v prvom kroku zhlukuje sekvencie na základe podobnosti (podobnosť určuje editačná vzdialenosť medzi sekvenciami). Pre každý zhluk potom vygenerujú najdlhší aproximatívny sekvenčný vzor, ktorý nazývajú vzor zhody (ang. *consensus pattern*). Každej sekvencií v zhluku sú pridané prázdne položky tak aby mali všetky sekvencie v rámci zhluku rovnakú dĺžku a tak aby vzniklo tzv. globálne optimálne zarovnanie sekvencií (ang. *global sequence alignment*). To znamená, že suma editačných vzdialenosti medzi všetkými dvojicami sekvencií je minimalizovaná. Z takto zarovnaných sekvencií je odvodená váhovaná sekvencia položiek, ktorá sumarizuje počty výskytov jednotlivých položiek v zarovnaných sekvenciách. Tento vzor aj s pridanou informáciou o váhe jednotlivých položiek je kompaktnou reprezentáciou všetkých sekvencií v danom zhluku (Shen, Wang, & Han, 2014).

Hľadanie k-najlepších uzavretých vzorov

Ako sme to už viackrát spomenuli tak dolovanie uzavretých vzorov prináša výrazne zmenšenie prehľadávaného priestoru a pritom sa nestráca žiadna informácia, pretože možno odvodiť všetky ostatné vzory aj s ich hodnotou podpory. Čo sa týka väčšiny algoritmov dolovania frekventovaných vzorov je problematické najmä vhodné nastavenie hladiny minimálnej podpory. Ak je príliš vysoká nemusia sa nájsť dôležité vzory a ak je príliš malá môže dôjsť k priveľkému výpočtovému zaťaženiu. Tento problém rieši algoritmus *TSP* navrhnutý v práci (Tzvetkov, Yan, & Han, 2005). Namiesto vstupného parametra minimálnej podpory vyžaduje ako vstupný parameter minimálnu dĺžku vzorov a *k* čo je požadovaný počet uzavretých vzorov, ktoré majú byť nájdené. Algoritmus veľmi skoro dokáže nájsť uzavreté vzory s najvyššou podporou a postupne hľadať ďalšie. (Shen, Wang, & Han, 2014)

3.3 Iné reprezentácie vzorov správania

V tejto časti opisujeme niektoré vybrané ďalšie zaujímavé prístupy k reprezentácií vzorov správania používateľov a ich získavaniu.

3.3.1 Grafová reprezentácia

V práci (Jalali, Mustapha, Nasir Sulaiman, & Mamat, 2010) zhlukujú používateľské sedenia podľa ich podobných vlastností. Z týchto zhlukov potom vytvárajú tzv. navigačné vzory, čo je v podstate reprezentácia vzorov správania používateľov. Opisovaný proces zhlukovania využíva algoritmus pozostávajúci z týchto základných krokov:

1. Pre každý pár stránok sa počíta stupeň prepojenia na základe vzorca uvažujúceho frekvencie spoločného výskytu týchto stránok v používateľských sedeniach a časového rozdielu medzi prístupmi k jednotlivým stránkam v rámci daného sedenia.

$$W_{a,b} = \frac{2\,TC_{ab}\,FC_{ab}}{TC_{ab} + FC_{ab}}$$

   - $W_{a,\,b}$ je stupeň prepojenia medzi stránkami *a* a *b* reprezentovaný ako harmonický priemer časového prepojenia a frekvencie spoločného výskytu stránok v používateľskom sedení.
   - $FC_{a,\,b}$ meria frekvenciu výskytu stránok *a* aj *b* v každom sedení.

$$FC_{a,b} = \frac{N_{ab}}{\max\{N_a, N_b\}}$$

   - N_{ab} je počet sedení obsahujúcich stránku *a* aj *b*. N_a a podobne aj N_b je počet sedení obsahujúcich len stránku *a* resp. stránku *b*.
   - $TC_{a,\,b}$ je stupeň časového prepojenia medzi návštevami každých dvoch stránok webového sídla.

$$TC_{ab} = \frac{\sum_{i=1}^{N} \frac{T_i}{T_{ab}} \frac{f_a(k)}{f_b(k)}}{\sum_{i=1}^{N} \frac{T_i}{T_{ab}}}$$

- T_i je časová dĺžka trvania *i*-teho sedenia, ktoré obsahuje návštevy oboch stránok *a* aj *b.*
- T_{ab} je časový rozdiel medzi časmi navštívenia stránok *a* a *b* v rámci *i*-teho sedenia.
- Jedna z možných reprezentácií funkcie f je $f_a(k) = k$ ak sa stránka *a* nachádza na k-tej pozícii v rámci sedenia. Ak by sme napríklad chceli zvýšiť dôležitosť pozície stránky v rámci sedenia môžeme použiť f kde $f(k) = k^2$.

2. Vytvorenie neorientovaného grafu reprezentovaného maticou susedností. Počet hrán je obmedzený danými konštantnými hraničnými hodnotami.
3. Algoritmy delenia grafu sú využité na rozdelenie grafu na k oddelených častí (vzorov správania) obsahujúcich silno prepojené stránky pričom spojenia medzi identifikovanými komponentami sú slabé.

Získané vzory správania ďalej používajú na generovanie odporúčaní podľa aktuálneho správania používateľov. Vzory relevantné k aktuálnemu správaniu používateľa hľadá pomocou algoritmu LCS (ang. *longest common subsequence*).

3.3.2 Frekventované super-sekvencie

V práci (Yu & Korkmaz, 2015) skúmajú novú formu hľadania frekventovaných vzoroch v sekvenčných dátach, ktorú nazývajú dolovanie frekventovaných super-sekvencií (ang. *super-sequence frequent pattern mining*). Všetky doposiaľ uvedené algoritmy sa dajú použiť na hľadanie frekventovaných subsekvencií zo sekvencií, teda hľadanie častí týchto sekvencií, ktoré sa vyskytujú v mnohých sekvenciách. Super-sekvencie sú také sekvencie, ktoré môžu obsahovať spoločné časti z viacerých sekvencií, teda spájajú viaceré nájdene frekventované subsekvencie. Super-sekvencie odhaľujú skryté štruktúry ukryté medzi viacerými sekvenciami v databáze. Autori uvádzajú analógiu tohto problému s problémom hľadania najťažšej cesty v grafe. Tento graf si možno predstaviť ako vážený orientovaný graf, kde uzly predstavujú navštívené stránky webového sídla a hrany medzi dvoma uzlami vzniknú ak existuje taká po sebe idúca dvojica navštívených stránok v niektorom zo sedení. Váha danej hrany predstavuje počet takýchto dvojíc. Uvedený problém hľadania najťažšej cesty v grafe je známy ako *NP-zložitý.* Autori v tejto práci navrhli heuristiku, ktorá pomocou techník dynamického programovania pomáha efektívnejšie riešiť tento problém.

3.3.3 Zhlukovanie sedení používateľov

V práci (Vellingiri, Kaliraj, Satheeshkumar, & Parthiban, 2015) používajú zhlukovanie dát na získanie vzorov správania. Používajú špeciálny váhovaný fuzzy pravdepodobnostný c-means zhlukovací algoritmus (FPCM) na zhlukovanie sedení používateľov. Tento generuje priradenie používateľských sedení do zhlukov s rôznymi pravdepodobnosťami. Nájdené zhluky (ich centrá) predstavujú vzory správania používateľov. V tejto práci ďalej používajú nájdené vzory na predikciu správania používateľov podľa ich posledných vykonaných akcií v sedení. Na klasifikáciu správania používateľov do správnych vzorov používajú neuro-fuzzy inferenčný systém ANFIS-SA, ktorý v predikcií výrazne prekonáva napr. jednoduchší algoritmus LCS (ang. *longest common subsequence*).

3.3.4 Markovovské modely a Kohonenove sebaorganizujúce mapy (SOM)

Prístupy k získavaniu vzorov správania sú skutočne rôznorodé o čom svedčia i ďalšie dve práce.

V práci (Makker & Rathy, 2011) kombinujú tri metódy dolovania z dát: zhlukovanie, markovovské modely a asociačné pravidlá. Sedenia používateľov najskôr zhlukujú pomocou algoritmu *k-means.* Pre každý získaný zhluk počítajú jednoduchý Markovovský model (markovovský reťazec – ang. *markov chain*) umožňujúci jednoduchú predikciu. Markovovský reťazec predstavuje určenie pravdepodobnosti výskytu ďalších akcií používateľa na základe jeho *k* posledných akcií. Táto pravdepodobnosť sa počíta na základe histórie všetkých známych postupností akcií v danom zhluku. Markovovský reťazec s nízkym stupňom *k* prináša pri predikcií vysoké pokrytie. Okrem toho počítajú zo vstupných dát asociačné pravidlá. Samotná predikcia je potom kombináciou výsledkov markovovského reťazca s vysokým pokrytím a asociačných pravidiel, ktoré dosahujú vysokú presnosť.

V práci (Etminani, Delui, Yenehsari, & Rouhani, 2009) používajú na extrakciu zaujímavých vzorov tzv. Kohonenove sebaorganizujúce mapy (SOM). SOM je typom neurónovej siete. Trénovacia fáza SOM však nevyžaduje určenie cieľového vektoru dát, takže sa jedná o učenie bez učiteľa. SOM je možné použiť ako zhlukovací algoritmus. Jeho výhodou je, že nevyžaduje definovať počet požadovaných zhlukov ako napr. *k-means.* SOM je založená na súťaživom princípe, výstupné neuróny siete medzi sebou súťažia, ktorý je najbližšie vstupnej dátovej inštancií. Víťazný neurón a jeho susedné neuróny upravia svoje váhy tak aby sa priblížili vstupným dátam. Pomocou SOM teda zhlukujú sedenia používateľov a váhované centrá týchto zhlukov predstavujú získané vzory správania. Výhodou tohto prístupu je, že môžu byť nájdené relevantné vzory rôznych dĺžok a aj s nižšou frekvenciou výskytov (vďaka použitému typu zhlukovania).

3.4 Sumarizácia

V úvode tejto kapitoly sme definovali pojem vzory správania, ktoré v tejto práci chápeme ako v literatúre bežne uvádzané frekventované vzory (ang. *frequent patterns*). Frekventované vzory možno rozdeliť na frekventované sekvencie a frekventované množiny podľa toho či zachovávajú sekvenčnú postupnosť položiek alebo nie.

Definovali sme problematiku hľadania frekventovaných množín a uviedli niekoľko existujúcich algoritmov, ktoré túto problematiku riešia. V prílohe A tiež uvádzame podrobnejší opis jednotlivých algoritmov. Riešime tam tiež aké problémy s výpočtovou náročnosťou má prvý navrhnutý algoritmus v tejto problematike s názvom *Apriori.* Najmä pri generovaní kandidátov na frekventované množiny a počítaní ich podpory, čo si vyžaduje viacero prechodov databázou transakcií. Ďalšie prístupy sa snažia tieto problémy vyriešiť ako napr. algoritmus *FP-Growth*, ktorý doluje frekventované množiny bez potreby generovania kandidátnych frekventovaných množín pomocou kompaktnej dátovej štruktúry s názvom *FP-Tree*. Napr. algoritmus *CHARM* ešte výraznejšie zmenšuje problémový priestor vďaka tomu, že hľadá len uzavreté frekventované množiny, ktoré kompaktne reprezentujú väčšie množstvo inak redundantných vzorov.

Ďalej sme definovali problematiku sofistikovanejšieho spôsobu reprezentácie a dolovania frekventovaných vzorov v podobe frekventovaných sekvencií. V prílohe B tiež bližšie opisujeme niektoré algoritmy. Opísali sme tiež niektoré ďalšie zaujímavé algoritmy a prístupy, ktoré rozširujú základné prístupy k dolovaniu frekventovaných sekvencií a snažia sa vyriešiť niektoré problémy s ktorými sa v praxi stýkajú. Na konci tejto kapitoly sme tiež uviedli niekoľko ďalších zaujímavých prístupov k reprezentácií vzorov správania a prístupov k ich získavaniu napr. pomocou zhlukovania, markovovských modelov či algoritmov rozdeľovania grafu.

Dôležitou výzvou v oblasti dolovania frekventovaných vzorov je výpočtová náročnosť samotnej úlohy. Už aj pre stredne veľké datasety je samotný priestor prehľadávania obrovský a rastie exponenciálne s dĺžkou transakcií v datasete (Aggarwal, Bhuiyan, & Hasan, 2014). Opisované algoritmy sa zameriavajú na čo najlepšie riešenie práve čo sa týka zmenšovania prehľadávaného priestoru a zmenšovania výpočtovej náročnosti problému. Tu sa otvárajú aj ďalšie výzvy ako riešiť tento problém efektívnejšie a flexibilne pre rýchle prúdy dát, čomu sa venujeme v ďalšej kapitole tejto práce.

4 Vzory správania v prúdoch dát

V úvode kapitoly 3 sme už uviedli, že v tejto práci pod pojmom vzory správania rozumieme v literatúre bežne uvádzane tzv. frekventované vzory, čo môžu byť buď frekventované množiny alebo frekventované sekvencie získané z dát zachytávajúcich správanie používateľov webového sídla. V tejto kapitole prenesieme tento pojem do oblasti spracovania prúdu dát, kde sa vynárajú nové problémy a výzvy.

Získavanie informácií z prúdov dát je dôležitou interdisciplinárnou výskumnou oblasťou (Krempl, et al., 2014). V dnešnej dobe v súvislosti s neustálym rastom produkovaných dát pribúdajú aj výzvy na ich efektívne spracovanie. Známe vlastnosti, s ktorými sú spájané problémy spracovania veľkých dát, sú rýchlosť (ang. *velocity*) a objem (ang. *volume*). Metódy na spracovávanie prúdov dát sa musia vysporiadať s rýchlym pribúdaním nových dátových inštancií. Tie môžu pribúdať teoreticky do nekonečného objemu, ktorý by nebol spracovateľný bežnými metódami určenými pre statické dáta.

Algoritmy spracovania prúdov dát sú postavené na modeloch, ktoré sú inkrementálne aktualizované s prichádzajúcimi inštanciami. Tradičné metódy často vyžadujú viaceré prechody inštanciami v dátach ale v prúde dát je každá inštancia spracovaná práve raz. Počas vývoja dátového prúdu v čase môže nastávať k tzv. konceptuálnemu posunu. Konceptuálny posun je významným problémom v dolovaní informácií z prúdov dát a vyplýva zo zmien reálneho prostredia v ktorom dáta vznikajú. Faktory vyvolávajúce takéto zmeny sú často skryté a neznáme. Konceptuálny posun spôsobuje vážny problém pri modeloch vytvorených z neaktuálnych dát, ktoré sú samozrejme v takom prípade nesprávne. Konceptuálny posun môžeme klasifikovať podľa rýchlosti zmien ako náhly alebo postupný. Metóda spracovania prúdových dát, by mala takéto zmeny vedieť identifikovať, odlíšiť ich od bežného šumu v dátach a adaptovať sa čo najrýchlejšie (Tsymbal, 2004). Napr. pri dolovaní frekventovaných množín z prúdu dát môže konceptuálny posun celkom zmeniť frekventované množiny aké sa v dátach vyskytujú. K týmto zmenám môže dokonca dochádzať sezónne (napríklad často spolu nakupované položky v internetovom obchode s oblečením budú celkom iné v zime ako v lete).

Ako sme už v kapitole 3 uviedli, v tejto práci sa orientujeme na vzory správania reprezentované ako frekventované vzory v dátach o správaní používateľov webového sídla. V súvislosti s úlohou dolovania frekventovaných vzorov v prúde dát sa stretávame najmä s týmito troma problémami (Lee, Jin, & Agrawal, 2014):

- Exponenciálny nárast počtu prehľadávaných vzorov. Napr. algoritmus pre statické dáta *Apriori* spracuje $O(k^2)$ podmnožín aby našiel jeden vzor o dĺžke k.
- Pamäťová náročnosť kvôli veľkému priestoru prehľadávania.
- Potreba balansovania požiadaviek na presnosť a požiadaviek na efektívnosť. Spresňovanie výsledkov znamená viac použitej pamäte a viac výpočtového času. Používateľovi by teda malo byť umožnené nastaviť tento balans podľa jeho aktuálnych potrieb.

V časti 4.1 sa venujeme podrobnejšie problematike dolovania frekventovaných množín z prúdu dát a bližšie opisujeme vybrané algoritmy. V časti 4.2 trocha odbočujeme od

tematiky hľadania vzorov správania a opisujeme možnosti zhlukovania v prúde dát. Tejto problematike sa venujeme z dôvodu, že jedným z cieľov tejto práce je hľadanie špecifických vzorov správania pre skupiny používateľov. Tieto skupiny môžu byť identifikované práve algoritmami na zhlukovanie v prúde dát. Okrem toho, ako sme uviedli aj v niektorých príkladoch v časti 3.3, zhluky používateľských sedení môžu byť tiež použité ako reprezentácia vzorov správania. V prílohe E ešte dodatočne analyzujeme a opisujeme existujúce rámce použiteľné pre spracovanie prúdu dát.

4.1 Dolovanie frekventovaných množín v prúde dát

V časti 3.1 sme uviedli definíciu úlohy dolovania frekventovaných množín v statických dátach. V tejto časti sa venujeme algoritmom, ktoré dolujú frekventované množiny z prúdu dát.

Rýchlosť pribúdania nových dátových inštancií v prúde dát vylučuje možnosť ukladania ich celej histórie do pamäte a opätovné prechádzanie. Ani ukladanie väčších častí prúdu dát do pamäte nie je typicky vhodným riešením (Calders, Dexters, Gillis, & Goethals, 2014). Na vysporiadanie sa s týmto problémom bolo navrhnutých niekoľko prístupov. Väčšinou sú založené na princípe hľadania frekventovaných množín v rámci okna obsahujúceho posledných *w* transakcií. Toto okno môže byť (Lee, Jin, & Agrawal, 2014):

- **Míľnikové okno** (ang. *landmark window*). $W = < T_1, T_2, \dots, T_t >$, kde T_i predstavuje transakcie v jednej dávke. T_1 je najstaršia časť okna a T_t je najnovšia časť.
- **Posúvajúce sa okno** (ang. *sliding window*). $W = < T_{t-w+1}, \dots, T_t >$, kde *w* je veľkosť posúvajúceho sa okna.
    - S fixovaným časovým intervalom (ang. *time-sensitive*). Počet obsiahnutých transakcií *w* je rovný počtu transakcií, ktoré prišli v prúde dát za poslednú časovú jednotku t_w.
    - Pevne fixovanej dĺžky (ang. *transaction sensitive*), obsahujúce vždy rovnaký počet transakcií *w*.
- **Okno s tlmením** (ang. *damped window*). V podstate ide o míľnikové okno, kde váha (podpora) predchádzajúcich nájdených množín postupne klesá podľa tzv. faktoru tlmenia *d* (ang. *decay factor*), kde $0 < d <= 1$.
- **Model s oknami variabilnej dĺžky** (ang. *tilted-time window*). Špecifický model umožňujúci hľadať frekventované množiny nad množinou viacerých okien variabilnej dĺžky. Každé okno môže byť napríklad dvakrát také dlhé ako predchádzajúce (aktuálnejšie) okno.

Prístupy hľadania frekventovaných množín možno ďalej klasifikovať podľa toho koľko transakcií je vstupom do procedúry aktualizácie aktuálnych frekventovaných množín (Quadrana, Bifet, & Gavaldà, 2015):

- Prístupy s aktualizáciou v dávkach,
- Prístupy s aktualizáciou v každej transakcií.

A tiež podľa toho či výstupom sú:

- exaktné frekventované množiny (teda také aké by našli aj statické metódy),
- alebo aproximatívne frekventované množiny.

Podľa (Quadrana, Bifet, & Gavaldà, 2015) prístupy s aktualizáciou v dávkach prinášajú vyššiu rýchlosť spracovania transakcií za cenu možnej dočasnej straty presnosti aktuálnej udržiavanej množiny frekventovaných množín. Hľadanie exaktných frekventovaných množín v prúde dát môže byť nezvládnuteľné z hľadiska pamäťovej a výpočtovej náročnosti keďže v takom prípade je nutné udržiavať v pamäti všetky nefrekventované množiny od začiatku spracovania prúdu (Quadrana, Bifet, & Gavaldà, 2015). Na zmenšenie prehľadávaného priestoru a tým pádom aj zrýchlenie spracovania prúdu dát je vhodné použiť algoritmus dolovania uzavretých frekventovaných množín alebo maximálnych frekventovaných množín. Uzavreté frekventované množiny (3.1) sú komprimovanou, kompletnou a neredundantnou reprezentáciou všetkých frekventovaných množín. Maximálne frekventované množiny (3.1) sú komprimovanou a neredundantnou reprezentáciou frekventovaných množín pri ktorej sa však stráca informácia o podpore niektorých frekventovaných množín.

4.1.1 Algoritmy dolovania frekventovaných množín

V tejto časti opisujeme základné charakteristiky vybraných známych algoritmov dolovania frekventovaných množín v prúde dát.

Jeden z prvých algoritmov na inkrementálne dolovanie uzavretých frekventovaných množín *MOMENT* bol navrhnutý v práci (Chi, Wang, Yu, & Muntz, 2004). Tento algoritmus hľadá exaktné frekventované množiny, využíva posúvajúce sa okno a frekventované množiny aktualizuje po každej transakcií. Informácie o všetkých množinách sa ukladajú do špeciálnej stromovej dátovej štruktúry *CET* (ang. *closed enumeration tree*), ktorá rozdeľuje do rôznych typov uzlov nefrekventované množiny, potencionálne frekventované množiny a uzavreté frekventované množiny. Na ukladanie informácií o všetkých transakciách v aktuálnom okne používa dátovú štruktúru *FP-tree*, podobnú ako v algoritme *FP-Growth* (Han, Pei, & Yin, 2000), ale bez orezávania nefrekventovaných množín. To, že algoritmus *MOMENT* musí ukladať všetky nájdené množiny (aj nefrekventované) aj napriek uvedeným kompaktným štruktúram spôsobuje značné zaťaženie pamäte (Quadrana & Mestre, 2012). Podrobnejšie tento algoritmus opisujeme v prílohe C.

Algoritmus *NEWMOMENT* (Li, Ho, Kuo, & Lee, 2006) je vylepšením algoritmu *MOMENT*. Reprezentuje transakcie a množiny pomocou bitových sekvencií. Pohyb okna uskutočňuje pomocou bitovej operácie ľavého posunu. Počítanie podpôr množín je vykonávané pomocou bitovej operácie súčinu. Bitová reprezentácia týchto štruktúr zefektívňuje a zlepšuje výkonnosť oproti pôvodnému algoritmu *MOMENT* (Quadrana & Mestre, 2012).

Ďalším algoritmom, ktorý využíva odlišný prístup a iné dátové štruktúry na udržanie informácie o uzavretých množinách je *CLOSTREAM* (Yen, Wu, Lee, Tseng, & Hsieh, 2011). Hľadá všetky exaktné uzavreté množiny, využíva posúvajúce sa okno a aktualizáciu po každej transakcií. Používa nové dátové štruktúry *ClosedTable*, *CidList* a funkciu *SET*

na hľadanie uzavretých množín podobných k aktuálnej prichádzajúcej transakcií. Oproti predchádzajúcim prístupom nepotrebuje zložito prehľadávať stromovú štruktúru s množinami, ale pomocou prieniku aktuálnej transakcie so špecifickými uzavretými množinami nachádza množiny, ktoré majú byť aktualizované. Na zefektívnenie tohto hľadania využíva dve dočasné hašovacie tabuľky. Podrobnejšie ho opisujeme v prílohe C.

Algoritmus *INSTANT* (Mao, Wu, Zhu, Chen, & Liu, 2007) hľadá exaktné maximálne množiny v prúde dát s využitím míľnikového okna. Ukladá množiny do osobitných polí podľa počtu ich výskytov až po hranicu minimálnej podpory. Každá nová transakcia je porovnaná s existujúcimi a prieniky sú uložené do správneho poľa. Ďalej v práci (Li, Zhang, & Chen, 2012) autori vylepšujú výpočtovú a pamäťovú efektívnosť tohto algoritmu pomocou kompaktnej dátovej štruktúry s názvom *FP-FOREST* vychádzajúcej zo známej dátovej štruktúry *FP-tree*.

Hľadanie exaktných frekventovaných množín kladie na algoritmy priveľké pamäťové nároky. Takéto algoritmy majú tiež vážny problém s adaptáciou na konceptuálne posuny v prúde dát. Tieto problémy môžu vyriešiť algoritmy, ktoré hľadajú aproximácie frekventovaných množín a ich podpôr. Odstúpenie od požiadavky hľadať exaktné frekventované množiny má za následok rapídne zlepšenie výkonnosti a zníženie pamäťových nárokov, okrem toho môže byť v mnohých aplikáciách takáto reprezentácia dostatočná a vie sa lepšie vysporiadať aj s konceptuálnym posunom (Quadrana & Mestre, 2012).

V práci (Song, Yang, Cui, Liu, & Xie, 2007) navrhujú algoritmus *CLAIM*. Je to algoritmus na dolovanie aproximácie uzavretých frekventovaných množín z prúdov dát, s aktualizáciou po každej transakcií nad posúvajúcim sa oknom. Tento algoritmus sa snaží riešiť problém keď časté malé konceptuálne posuny v prúde dát môžu zbytočne spomaľovať algoritmus redefiníciou pojmu uzavretá frekventovaná množina a určením intervalov hodnôt podpory, ktoré sú považované za rovnaké.

Dolovaniu aproximatívnych frekventovaných množín sa venujú v práci (Chang & Lee, 2003). Navrhujú algoritmus *estDec* používajúci okno s tlmením. Počty výskytov tzv. signifikantných množín sú uchovávané prostredníctvom lexikografickej stromovej štruktúry (ang. *prefix-tree*). Množiny sú signifikantné iba ak ich aktuálna podpora je vyššia ako používateľom definovaná hranica (nižšia ako minimálna podpora). V práci (Shin, Lee, & Lee, 2014) vylepšujú algoritmus *estDec* o možnosť menej presnej ale kompaktnejšej reprezentácie množín. Navrhujú algoritmus *estDec+* so štruktúrou *CP-tree* (ang. *Compressible Prefix-Tree*). Viacero uzlov, ktoré by boli inak samostatné v pôvodnom prefixovom strome, je zlúčených ak ich rozdiel v podpore je menší ako definovaná hranica δ. Pomocou nastavovania δ a miery tlmenia d možno nastaviť úroveň kompresie frekventovaných množín do kompaktnejšej reprezentácie (to je rozdiel oproti uzavretým a maximálnym frekventovaným množinám). Tento algoritmus podrobnejšie opisujeme v Prílohe C.

IncMine (Cheng, Ke, & Ng, 2008) je aproximatívny algoritmus hľadania uzavretých frekventovaných množín z rýchlych prúdov dát nad posúvajúcim sa oknom. Ako jediný z opisovaných algoritmov uskutočňuje aktualizáciu množín v dávkach. To prináša vyššiu rýchlosť spracovania transakcií za cenu možnej dočasnej straty presnosti udržiavanej

množiny frekventovaných množín (Quadrana, Bifet, & Gavaldà, 2015). Okrem klasickej hladiny minimálnej podpory používa aj uvoľnenú hladinu minimálnej podpory, ktorá zabezpečuje zotrvanie tzv. potencionálne frekventovaných uzavretých množín v okne (ang. *semi-frequent closed itemsets*). Na zefektívnenie hľadania množín navrhujú autori použitie invertovaného indexu. Autori algoritmu experimentálne potvrdzujú jeho efektivitu a rýchlosť ako i porovnateľnú presnosť a pokrytie v porovnaní s inými algoritmami (*MOMENT* a *LCSW*). Tento algoritmus podrobne opisujeme v Prílohe C.

4.2 Zhlukovanie v prúde dát

V tejto podkapitole opisujeme niekoľko základných princípov a algoritmov zhlukovania v prúde dát. Problém zhlukovania je chápaný ako hľadanie množiny zhlukov (kategórií) objektov, tak že objekty patriace do rovnakého zhluku sú navzájom viac podobné ako objekty v rôznych zhlukoch (Silva, et al., 2013). Klasické algoritmy zhlukovania sú predmetom výskumu už dekády a ich analýza je mimo rozsah tejto práce. Zhlukovanie v prúde dát je náročnejšia úloha pretože kladie na algoritmy nové požiadavky (Silva, et al., 2013):

- Kompaktnosť reprezentácie zhlukov. Je dôležité aby veľkosť reprezentácie nerástla príliš rýchlo s prichádzajúcimi dátovými inštanciami.
- Rýchle a inkrementálne spracovanie nových dátových inštancií.
- Rýchla adaptácia na meniacu sa dynamiku dát. Detekcia nového rozloženia dát (niektoré zhluky vznikajú a iné zanikajú).
- Jasné a rýchle identifikovanie odľahlých dátových inštancií.

Podľa (Silva, et al., 2013) najbežnejší prístup zhlukovania v prúde dát, ktorý implementuje viacero algoritmov, pozostáva z dvoch hlavných krokov:

- Dátová abstrakcia (alebo aj *online komponent*).
- Makrozhlukovanie (alebo aj *offline komponent*).

Online komponent zabezpečuje rýchlu transformáciu dát z prúdu do pamäťovo efektívnych dátových štruktúr. Tieto dátové štruktúry sú sumarizáciou a aproximáciou aktuálneho stavu dát v prúde. Podobne ako pri algoritmoch dolovania frekventovaných vzorov v prúde dát používajú algoritmy zhlukovania rôzne prístupy na úpravu váh dátových inštancií podľa ich časovej aktuálnosti. Populárne prístupy používajú časové okná (4.1): míľnikové okno, posúvajúce sa okno, okno s tlmením. Offline komponent zabezpečuje na požiadanie pohľad na rozdelenie dát do rôzne veľkých zhlukov. Umožňuje používateľovi skúmať zhluky v dátovom prúde v rôznych časových momentoch. Vstupom do tohto kroku je aktuálna sumarizácia dát vytvorená online komponentom, vďaka čomu je aj tento krok značne efektívny. Na hľadanie zhlukov v týchto dátových štruktúrach môžu byť použité klasické dávkové zhlukovacie algoritmy ako napr. *k-means* alebo *DBSCAN*.

4.2.1 Algoritmy zhlukovania

Bolo navrhnutých viacero algoritmov zhlukovania v prúde dát. Prvé prístupy hľadali zhluky pre celý dátový prúd (od začiatku po koniec), boli teda len jednoprechodovými

algoritmami. Také prístupy sú však nepoužiteľné v prípade prúdov, ktoré predstavujú potencionálne nekonečný proces pribúdania dát a s časom sa vyvíjajú (Silva, et al., 2013).

Clustream (Aggarwal C. C., Han, Wang, & Yu, 2003) je založený na online mikrozhlukovacom komponente, ktorý rýchlo spracováva a transformuje prichádzajúce dátové transakcie do kompaktnej aproximatívnej štatistickej reprezentácie (mikrozhlukov), a offline makrozhlukovacom komponente, ktorý z tejto reprezentácie na požiadanie získa finálne zhluky (makrozhluky). Rýchla online transformácia vstupných dát do podoby mikrozhlukov umožňuje rýchlu reakciu na zmeny v dátach. Vďaka možnosti nastavenia konštantného počtu mikrozhlukov udržiavaných v pamäti je *CluStream* dobre škálovateľný (k veľkosti prúdu, počtu zhlukov, dimenzionalite dát). Vďaka kompaktnej reprezentácií dát pomocou mikrozhlukov je offline proces hľadania makrozhlukov značne urýchlený. Makrozhluky sú získavané pomocou modifikovaného *k-means* algoritmu. Vo fáze inicializácie *k-means* sa nevyberajú počiatočné body náhodne ale mikrozhluky s vyšším počtom bodov sú vybrané pravdepodobnejšie. Podrobnejšie *CluStream* opisujeme v Prílohe D.

Podobný prístup je použitý aj v ďalších prácach s použitím rozličných makrozhlukovacích algoritmov a prístupov k získavaniu a reprezentácií mikrozhlukov. Napr. algoritmus zhlukovania podľa hustoty dát *DenStream* (Cao, Ester, Qian, & Zhou, 2006), používa ako makrozhlukovací algoritmus *DBSCAN* a zavádza nové koncepty tzv. jadrových a mimoležiacich mikrozhlukov (ang. *core* resp. *outlier microclusters*). *DenStream* tak dokáže detekovať zhluky ľubovoľných tvarov a nevyžaduje od používateľa definovať počet zhlukov ako vstupný parameter. Autori algoritmu ho porovnávajú s algoritmom *CluStream*, ktorý nie je založený na hustote bodov a preto nevie efektívne odhaľovať zhluky ľubovoľných tvarov. *DenStream* teda v ich výsledkoch na testovaných datasetoch jednoznačné prevyšuje *CluStream* (aj viac ako o 50%) v kvalite nájdených zhlukov (miere zhody so skutočnosťou). Podrobnejšie *DenStream* opisujeme v Prílohe D.

Prístup z algoritmu *CluStream* je použitý tiež v algoritme *HPStream* (Aggarwal C. C., Han, Wang, & Yu, 2004). Tento algoritmus pomocou projekcie dát zhlukuje vysoko dimenzionálne dáta. Prekonáva algoritmus *CluStream* práve na vysoko dimenzionálnych prúdoch dát. Ďalším algoritmom s trochu odlišným prístupom, založený na hustote dát je *D-Stream* (Chen & Tu, 2007). Online komponent mapuje vstupné dáta do hustotnej mriežky (ang. *density grid*). Offline komponent zabezpečuje zhlukovanie dát v tejto mriežke. Používa techniku tlmenia (ang. *decaying technique*) na lepšie zachytenie dynamických zmien v zhlukoch.

ClusTree (Kranen, Assent, Baldauf, & Seidl, 2011) je algoritmus bez vstupných parametrov (okrem parametra počet zhlukov pre *makrozhlukovanie*), ktorý je schopný sa automaticky adaptovať rýchlosti prúdu dát. Používa kompaktný a samo adaptujúci sa index na zachytávanie sumárnych informácií o dátach z prúdu. Podobne ako *D-Stream* používa techniku tlmenia starších dát na lepšie zachytenie dynamických zmien v zhlukoch. Autori porovnávajú algoritmus *ClusTree* s algoritmami *CluStream* a *DenStream.* Podľa ich výsledkov dokáže *ClusTree* nájsť porovnateľné výsledky v kvalite zhlukov pri viac ako 10 násobne vyššej rýchlosti. Podrobnejšie *ClusTree* opisujeme v Prílohe D.

4.3 Sumarizácia

V úvode tejto kapitoly sme opísali požiadavky, ktoré sú kladené na úlohu hľadania frekventovaných vzorov v prúde dát. Hovorili sme o požiadavkách na rýchlosť, pamäťovú efektívnosť a požiadavke na možnosť balansovať efektívnosť výpočtu voči presnosti výsledkov. Prístupy k samotnému dolovaniu frekventovaných množín sú väčšinou založené na princípe hľadania frekventovaných množín v rámci okna obsahujúceho posledných *w* transakcií. Možno ich ďalej rozdeliť podľa toho či je toto okno fixnej dĺžky alebo predstavuje fixný časový interval, či aktualizujú frekventované množiny v dávkach alebo s každou prichádzajúcou transakciou, či hľadajú exaktné frekventované množiny alebo len aproximatívne frekventované množiny. Dolovanie aproximatívnych frekventovaných množín s aktualizáciou v dávkach vedie k rýchlejšiemu a pamäťovo efektívnejšiemu spracovaniu. V Prílohe C uvádzame podrobnejšie opisy vybraných algoritmov.

Opísali sme aj problém zhlukovania v prúde dát a niektoré algoritmy. Venovali sme sa mu kvôli tomu, že významnou súčasťou metódy navrhovanej v kapitole 6 bude práve komponent zabezpečujúci zhlukovanie modelov používateľa. Okrem toho, ako sme uviedli aj v časti 3.3, zhluky sedení používateľov môžu byť tiež považované za reprezentáciu vzorov správania. Najbežnejší prístup zhlukovania v prúde dát, ktorý implementuje viacero algoritmov, pozostáva z online a offline fázy. V online fáze sa vstupné dáta rýchlo transformujú do kompaktnej reprezentácie. V offline fáze sa na požiadanie z tejto kompaktnej reprezentácie získajú aktuálne zhluky. Opísali sme vybrané algoritmy pre zhlukovanie v prúde dát. V Prílohe D uvádzame ich podrobnejšie opisy.

V prílohe E sa ešte v krátkosti venujeme analýze existujúcich rámcov pre spracovanie prúdov dát.

5 Ciele práce

Porozumenie správaniu používateľov webového sídla je podstatné pri úlohách personalizácie či adaptácie jeho štruktúry a obsahu. Každý používateľ je unikátny a jeho akcie sú ovplyvnené jeho aktuálnym cieľom, kontextom, predchádzajúcimi znalosťami a ďalšími atribútmi. Keď sa však pozrieme na správanie viacerých používateľov spoločne, môžeme pozorovať viacero, v niektorých situáciách často opakovaných typických akcií, množín či sekvencií akcií. Tieto typické situácie nazývame vzory správania. Opísali sme viacero možných reprezentácií vzorov správania: napr. frekventované množiny a frekventované sekvencie (maximálne, uzavreté).

Uviedli sme viacero príkladov metód hľadania a aplikácie vzorov správania používajúcich rôzne reprezentácie vzorov, prístupy k ich získavaniu a použitia na predikciu či odporúčanie. Spoločným prvkom všetkých týchto metód (a pravdepodobne viacerých ďalších) je, že pozostávajú z výpočtovo náročnej offline časti vykonávajúcej trénovanie modelu (hľadanie vzorov), ktorý je následne použitý v online prostredí na odporúčanie či predikciu. Problém vzniká keď sa správanie používateľov mení a nájdené vzory sa stávajú neaktuálnymi a nepoužiteľnými.

Našim cieľom je preto navrhnúť metódu, ktorá bude schopná vykonávať hľadanie vzorov správania používateľov webového sídla aj ich aplikáciu v online čase a bude tiež schopná adaptovať sa na konceptuálne zmeny v správaní používateľov.

Analyzovali sme možné algoritmy hľadania frekventovaných vzorov dávkovým spôsobom v statických dátach. Dávkové prístupy často vyžadujú viacero prechodov cez dátové inštancie v databáze, čo výrazne znižuje ich použiteľnosť v online prostredí. V dnešnej dobe je totiž generovaných čoraz viac dát, zvlášť vo veľkých webových sídlach ako spravodajské portály, či sociálne siete. Tieto rýchlo pribúdajúce dáta je vhodnejšie spracovať jednoprechodovo (ako prúd dát) a rýchlo a flexibilne reagovať na zmeny v nich.

Preto sme sa rozhodli zamerať na algoritmy dolovania frekventovaných vzorov spracovávajúce dáta ako prúd. Keďže je našim cieľom spracovať dáta čo najrýchlejšie rozhodli sme sa použiť efektívnu a jednoduchú reprezentáciu vzorov správania v podobe uzavretých frekventovaných množín.

Úloha dolovania frekventovaných množín je jednoduchšia než dolovanie frekventovaných sekvencií, či asociačných pravidiel. Okrem toho použijeme algoritmus dolujúci uzavreté frekventované množiny čo je kompaktná, neredundantná a úplná reprezentácia všetkých frekventovaných množín.

Znalosť o správaní používateľov (napríklad v podobe vzorov správania) má široké využitie v rôznych aplikáciách.

My sa pri aplikácií vzorov správania zameriame na úlohu odporúčania stránok používateľom, ktorú použijeme aj na nepriame vyhodnotenie kvality nájdených vzorov správania.
Vzory správania môžu byť aplikovateľné v skupinách používateľov rôznych veľkostí. Veríme, že dynamickou identifikáciou menších komunít používateľov s podobným

správaním (pomocou algoritmu zhlukovania v prúde dát) môžeme nájsť špecifické vzory správania, ktoré by neboli inak rozpoznané na globálnej úrovni. Použitím oboch typov vzorov správania a ich kombináciou dosiahneme zlepšenie kvality a použiteľnosti vzorov, čo v tejto práci vyhodnotíme prostredníctvom aplikácie na úlohu odporúčania.

6 Metóda hľadania vzorov správania v prúde dát

V tejto kapitole opisujeme návrh metódy na dolovanie vzorov správania v prúde dát reprezentujúcich akcie používateľov webového sídla. Špecifikom navrhovanej metódy je, že hľadá vzory správania v dvoch úrovniach:

- *globálne vzory* správania identifikované pre všetkých používateľov webového sídla,
- *skupinové vzory* typické pre dynamicky identifikované užšie skupiny používateľov webového sídla.

Navrhovaná metóda je novou metódou v oblasti dolovania z dát o používaní Webu (ang. *Web Usage Mining*), ktorá v porovnaní voči metódam, ktoré sme uviedli v časti 2.2 umožňuje spracovanie dát ako rýchleho a potencionálne nekonečného prúdu a dolovanie špecifickejších vzorov pre skupiny používateľov.

Prínos navrhovanej metódy spočíva v schopnosti rýchlo a flexibilne odhaľovať aktuálne záujmy používateľov webového sídla a zmeny ich záujmov v čase. A to nielen na globálnej úrovni ale aj v rámci užších skupín používateľov s podobným správaním. Nachádzané vzory správania majú viacero možných aplikácií (pozri časť 2.2) ako napr. použitie na predikciu, odporúčanie, prednačítavanie stránok do dočasnej pamäte, či celkové porozumenie správania používateľov webového sídla.

Navrhovaná metóda spracováva prúd dát používateľských sedení reprezentovaných ako množiny vykonaných akcií používateľa (6.1). Modely používateľa (6.1) obsahujú informácie o predchádzajúcom správaní používateľov. Na základe týchto modelov sú používatelia zhlukovaní do skupín, pre ktoré sú identifikované *skupinové vzory správania.* Modely používateľa sú aktualizované po každom sedení, vďaka čomu sú skupinové vzory správania nachádzané v aktuálnych skupinách a metóda je schopná reagovať na zmeny v správaní používateľov a zmeny v rozdeleniach používateľov do skupín.

Navrhovaná metóda pozostáva z dvoch hlavných krokov vykonávaných kontinuálne nad prúdom dát (podrobnejšie v 6.2):

1. Zhlukovanie používateľov do skupín podľa podobnosti ich správania.
2. Dolovanie skupinových a globálnych vzorov správania reprezentovaných ako uzavreté frekventované množiny.

Súčasťou návrhu metódy je aj návrh spôsobu kombinácie globálnych a skupinových vzorov správania pre účel ich aplikácie v úlohe odporúčania zaujímavých položiek používateľom (6.2).

Keďže metóda kombinuje algoritmus zhlukovania s algoritmom dolovania frekventovaných množín má pomerne veľký počet vstupných parametrov. Pre zjednodušenie ich optimalizácie by mali byť optimalizované postupne jednotlivé časti metódy. Preto sme vstupné parametre rozdelili podľa ich spoločného účelu do 4 kategórií (6.3).

6.1 Základné pojmy

V tejto časti objasňujeme niektoré základné pojmy a princípy navrhovanej metódy ako je formát vstupných dát (6.1.1), reprezentácia vzoru správania a výber algoritmu pre dolovanie vzorov (6.1.2). Opisujeme model používateľa a spôsob jeho použitia spolu s výberom algoritmu pre zhlukovanie (6.1.3).

6.1.1 Zdroje a formát spracovávaných dát

Vstupom do navrhovanej metódy sú dáta z webového sídla. Prúd dát je generovaný ako postupnosť transakcií. Transakcia je vygenerovaná vždy na konci sedenia používateľa. Jedna vstupná transakcia, predstavuje konkrétne, práve ukončené sedenie používateľa. Táto transakcia obsahuje:

- id používateľa,
- usporiadaný zoznam akcií používateľa v rámci sedenia.

Za akcie sa môžu považovať návštevy konkrétnych stránok alebo širších kategórií stránok. Ale môžu to byť aj iné akcie používateľa v systéme ako napr. pridanie/vymazanie položky v košíku internetového obchodu, pridanie/vymazanie/editácia komentára, hodnotenia k položke, kúpa položky a mnoho ďalších.

6.1.2 Reprezentácia vzoru správania a dolovanie vzorov

V navrhovanej metóde chápeme vzor správania *p* ako štruktúru obsahujúcu nasledujúce údaje:

- Skupinový identifikátor, pokiaľ ide o skupinový vzor správania (ozn. *p.gid* z ang. *pattern's group identificator*).
- Uzavretú frekventovanú množinu (3.1) (ozn. *p.fci* z ang. *pattern's frequent closed itemset*). To je uzavretá množina identifikátorov akcií používateľa, ktorá spĺňa podmienku minimálnej podpory nad aktuálnym pohybujúcim sa oknom v rámci prúdu dát.
- Váhu vzoru (ozn. *p.sup* z ang. *pattern's support*). To je hodnota podpory množiny (3.1) aproximovaná vzhľadom k aktuálnemu pohybujúcemu sa oknu v prúde dát.

Na zabezpečenie dolovania uzavretých frekventovaných množín sme sa rozhodli upraviť algoritmus *IncMine* (4.1.1) z nasledujúcich dôvodov:

- Je to aproximatívny algoritmus s veľmi dobrou presnosťou aproximácie podľa výsledkov experimentov uvedených v práci (Cheng, Ke, & Ng, 2008). Aproximácia prispieva k vyššej rýchlosti spracovania oproti exaktným algoritmom ako napr. *MOMENT* (Chi, Wang, Yu, & Muntz, 2004) a *CLOSTREAM* (Yen, Wu, Lee, Tseng, & Hsieh, 2011) a umožňuje lepšiu adaptáciu na zmeny v dátach.
- Hľadá uzavreté frekventované množiny, ktoré sú kompletnou a neredundantnou reprezentáciou všetkých frekventovaných množín, ktorá zároveň výrazne napomáha k zmenšeniu prehľadávaného priestoru a zrýchleniu algoritmu.
- Využíva posúvajúce sa okno (4.1), ktoré sa oproti míľnikovému oknu dokáže lepšie vysporiadať s konceptuálnym posunom v prúde dát.

- Jeho implementácia je dostupná v rámci knižnice MOA (Quadrana & Mestre, 2012).
- *IncMine* ako jediný z analyzovaných algoritmov aktualizuje frekventované množiny po dávkach transakcií. Ako autori implementácie (Quadrana & Mestre, 2012) pozorujú vo svojich experimentoch, zdá sa tento prístup vhodný. V prípade aktualizácie po každej transakcií sú aktualizácie príliš časté, čo môže viesť k zbytočne častému vyťaženiu výpočtových zdrojov. Prínos jedinej transakcie oproti väčšej dávke transakcií spolu je pritom často zanedbateľný.

6.1.3 Zhlukovanie a reprezentácia modelov používateľa

Jedným z cieľov navrhovanej metódy je hľadanie skupinových vzorov správania v prúde dát. Preto je dôležitým problémom, ktorý musí metóda riešiť, aj reprezentácia modelu používateľa. Tento model je potrebné navrhnúť s čo najmenšími nárokmi na pamäť a tiež zabezpečiť aby využitie pamäte na uloženie modelov nerástlo proporcionálne k počtu spracovaných dát.

Model používateľa *u* je štruktúra pozostávajúca z nasledujúcich častí:

- Unikátny identifikátor používateľa (ozn *u.id* z ang. *user identificator*).
- Rad s obmedzenou kapacitou obsahujúci zoznam akcií v rámci posledných sedení používateľa (ozn. *u.aq z* ang. *user's actions queue*).
- Identifikátor skupiny do ktorej bol používateľ zaradený pri poslednom makrozhlukovaní (ozn. *u.gid* z ang. *user's group identificator*).
- Počet nových sedení používateľa od poslednej aktualizácie identifikátora skupiny (ozn. *u.nsc* z ang. *user's new sessions count*).
- Číslo posledného makrozhlukovania v ktorom bol používateľovi priradený skupinový identifikátor (ozn. *u.lmid* z ang. *user's last macroclustering identificator*).

Na zhlukovanie modelov používateľa sme sa rozhodli použiť algoritmus *CluStream* (Aggarwal C. C., Han, Wang, & Yu, 2003) s makrozhlukovacím komponentom *k-means*. Keďže naša metóda musí vykonávať pravidelné makrozhlukovanie (čo je typicky úloha offline komponentu algoritmu) potrebujeme aby táto fáza bola čo najrýchlejšia. *K-means* je rýchly a jednoducho nastaviteľný algoritmus. Vďaka možnosti nastavenia konštantného počtu mikrozhlukov udržiavaných v pamäti je *CluStream* dobre škálovateľný (k veľkosti, dimenzionalite prúdu dát a počtu zhlukov). Vďaka konceptu mikrozhlukov dokáže efektívne a rýchlo reagovať na prekvapujúce zmeny v dátach, čo vyhovuje požiadavkám našej metódy. Okrem toho je to algoritmus z ktorého vychádzajú viaceré ďalšie algoritmy, čo nám tak umožňuje pripraviť prostredie pre ďalšie experimentovanie a porovnávanie s inými algoritmami (pozri 8.3). Veľkou výhodou je tiež dostupnosť jeho implementácie v rámci MOA.

Prípadné použitie algoritmu založeného na hustote dát (ako napr. *DenStream*, *D-Stream*) by si vyžadovalo zmenu návrhu metódy v ktorom sa momentálne ráta s pevne definovaným počtom skupín používateľov. Nastavenie algoritmu založenom na hustote dát v dynamickom prostredí si tiež vyžaduje komplikovanejšiu optimalizáciu. Parametre týchto

algoritmov sú citlivé a zásadne zmeny v dátach by mohli preto znížiť účinnosť algoritmu a vyžadovať dodatočnú optimalizáciu.

Vo svete webových sídel môže byť počet používateľov veľmi dynamický. Noví používatelia prichádzajú a starí odchádzajú. Navrhovaná metóda v pravidelných intervaloch aktualizuje zhluky používateľov (makrozhlukovanie) podľa ich aktuálneho správania. Pre každý model používateľa si pamätá len obmedzený počet akcií aby nedošlo k nadmernému zaťaženiu pamäte a postupnému spomaľovaniu spracovania.

Vstupom do zhlukovacieho algoritmu *CluStream* je nová dátová inštancia vygenerovaná na požiadanie z histórie uskutočnených akcií modelu používateľa (*u.aq*) v podobe vektoru frekvencií výskytov jednotlivých vykonaných akcií. Pomocou tejto inštancie sú aktualizované mikrozhluky.

Fáza makrozhlukovania sa vykonáva pravidelne po definovanom počte aktualizácií mikrozhlukov. *lmid* je globálne počítadlo makrozhlukovania, ktoré je inkrementované vždy po novom makrozhlukovaní. Modelu používateľa *u* je priradený identifikátor skupiny (*u.gid*) vždy len pri jeho prvom uskutočnenom sedení od posledného makrozhlukovania. Teda iba vtedy ak platí: $u.lmid < lmid$. Zároveň sa priradí $u.lmid := lmid$.

Aby sa proporcionálne k času behu metódy nezvyšovalo zaťaženie pamäte, metóda pravidelne odstraňuje neaktívne modely používateľa. Po každom makrozhlukovaní sa prechádza zoznam modelov používateľa a odstráni sa každý taký model používateľa *u* pre ktorý platí, že $tcdiff < lmid - u.lmid$, kde *tcdiff* (z ang. *threshold of clustering identifiers difference*) je vstupný parameter metódy.

Dôležitým problémom pri algoritmoch dolovania z dát býva tiež dimenzionalita dát. S narastajúcou dimenzionalitou rôznych akcií sa bude spomaľovať algoritmus pre zhlukovanie aj algoritmus hľadania uzavretých frekventovaných množín, ktoré sú súčasťou navrhovanej metódy. Preto je dôležité pri generovaní dát, ktoré sú vstupom metódy, rátať s týmto problémom a pokiaľ je to možné generovať dáta s nižšou dimenzionalitou.

6.2 Spracovanie sedení používateľov

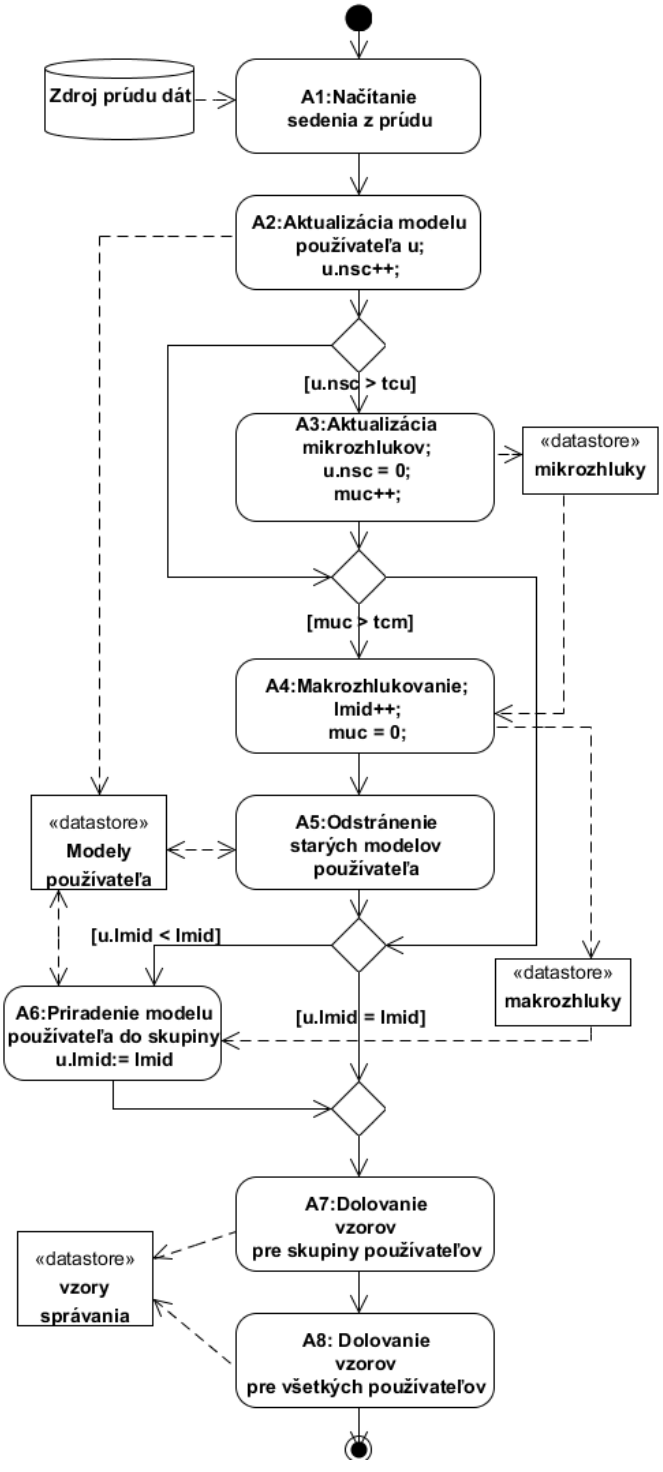
V tejto časti opisujeme samotný proces spracovania vstupných transakcií vstupujúcich do metódy (6.2.1), spôsob ich následnej aplikácie na úlohu odporúčania (6.2.2) a spôsob monitorovania a regulácie rýchlosti spracovania (6.2.3).

6.2.1 Proces

Celý proces spracovania jednej transakcie reprezentujúcej sedenie používateľa je znázornený na diagrame aktivít (Obrázok 1). V tejto časti bližšie opíšeme jednotlivé kroky tohto procesu:

1. **A1: Načítanie sedenia z prúdu dát.** Prvým krokom procesu spracovania sedenia používateľa reprezentovaného ako množina akcií je jeho načítanie zo zdroja prúdu dát.
2. **A2: Aktualizácia modelu používateľa *u*. Inkrementácia počtu zmien v modeli.** Do radu s obmedzenou kapacitou (atribút *u.aq*) obsahujúceho zoznam akcií vykonaných v rámci posledných sedení používateľa sa zaradí celé aktuálne sedenie

používateľa. Inkrementuje sa tiež počet sedení modelu používateľa od posledného priradeného makrozhlukovania (atribút *u.nsc* sa inkrementuje o 1).



Obrázok 1. Proces spracovania transakcie (sedenia používateľa) v navrhovanej metóde.

3. **A3: Aktualizácia mikrozhlukov. Inkrementácia počtu zmien v mikrozhlukoch.** Ak počítadlo zmien v modeli používateľa *u* (atribút *u.nsc*) dosiahlo hodnotu $u.nsc > tcu$, kde *tcu* je vopred definovaná hraničná hodnota počtu zmien v modeli používateľa (z ang. *threshold number of changes in usermodel*), tak sa počítadlo *u.nsc* vynuluje, mikrozhluky sú aktualizované pomocou dátovej inštancie

vygenerovanej z posledných akcií používateľa *u.aq* a počítadlo zmien v mikrozhlukoch *muc* (z ang. *microclusters' updates counter*) je inkrementované o 1.

4. **A4: Makrozhlukovanie.** Ak došlo k dostatočnému počtu zmien v mikrozhlukoch, teda ak $muc > tcm$ (*tcm* z ang. *threshold number of changes in microclusters*), tak sa z tejto štatistickej reprezentácie získajú tzv. makrozhluky. Na makrozhlukovanie je použitý váhovaný *k-means* algoritmus, ktorý zo vstupných mikrozhlukov hľadá vstupným parametrom *gc* (z ang. *groups count*) určený počet makrozhlukov. Inkrementuje sa tiež globálne počítadlo makrozhlukovaní *lmid* a vynuluje sa počítadlo zmien v mikrozhlukoch *muc.*
5. **A5: Odstránenie starých modelov používateľa.** Tento krok zabezpečuje, že pri každom novom makrozhlukovaní sa tiež odstránia staré modely používateľov, ktorí už dlhšiu dobu neboli aktívni. Odstránia sa tie modely, pre ktoré platí, že rozdiel medzi identifikátorom aktuálneho makrozhlukovania *lmid* a identifikátorom posledného makrozhlukovania v modeli používateľa *u* (atribút *u.lmid*) je väčší ako hodnota vstupného parametra *tcdiff* (z ang. *threshold of clustering id's difference*). Teda ak platí: *tcdiff* < *lmid – u.lmid*.
6. **A6: Aktualizácia priradenia modelu používateľa do skupiny.** Ak pre model používateľa *u* platí, že $u.lmid < lmid$, tak sa *u.lmid* nastaví na rovnakú hodnotu ako globálne *lmid* ($u.lmid := lmid$). Modelu používateľa *u* sa priradí identifikátor skupiny (atribút *u.gid*) podľa podobnosti vektora vygenerovaného z *u.aq* s vektormi centier aktuálnych makrozhlukov. Podobnosť je počítaná ako pearsonova korelácia medzi dvoma vektormi. Tá je vhodná na použitie pri riedkych vektoroch a zároveň je nezávislá na rôznom škálovaní hodnôt vo vektoroch.
7. **A7: Dolovanie vzorov správania pre skupiny používateľov.** Ak model používateľa *u* už má priradený identifikátor skupiny (atribút *u.gid*) tak je aktuálne sedenie vstupom do algoritmu hľadania uzavretých frekventovaných množín *IncMine*. Ten je upravený tak, že hľadá separátne uzavreté frekventované množiny pre jednotlivé skupiny používateľov identifikované zhlukovacím algoritmom. Pre každú skupinu je tak akoby simulovaný samostatný prúd len so sedeniami patriacimi danej skupine. Uzavreté frekventované množiny nájdené pre konkrétnu skupinu sú uložené do samostatnej dátovej štruktúry.
8. **A8: Dolovanie vzorov správania pre všetkých používateľov.** Tento krok sa vykoná s každým jedným sedením bez ohľadu nato či má alebo nemá priradený skupinový identifikátor. Aktuálne sedenie je vstupom do algoritmu hľadania uzavretých frekventovaných množín *IncMine*. Nájdené uzavreté frekventované množiny sa ukladajú do samostatnej dátovej štruktúry.

6.2.2 Aplikácia nájdených vzorov správania

Výstupom navrhnutej metódy je v každom čase množina objavených globálnych a skupinových vzorov správania. Úlohu hľadania vzorov správania môžeme považovať za samostatnú a oddelenú od úlohy ich aplikácie. Tieto úlohy teda samozrejme môžu byť realizované aj rôznymi komponentami architektúry konkrétneho softvérového riešenia. Vygenerované vzory správania môžu byť použité na rôzne účely ako napr. predikciu

správania používateľov, odporúčanie zaujímavých položiek používateľom, analýzu správania používateľov s rôznymi cieľmi ako napr. zlepšenie štruktúry a dizajnu webového sídla, podporu biznis rozhodnutí.

V tejto časti opisujeme jeden z možných spôsobov kombinácie skupinových vzorov s globálnymi a ich aplikácie v úlohe odporúčania zaujímavých položiek používateľom.

Dôležitým problémom pri generovaní odporúčaní je výber vzorov správania, ktoré čo najlepšie vystihujú súčasné správanie používateľa. Na určenie výstižných vzorov správania pre používateľa *u* a aktuálne okno vyhodnocovania W_e (vektor posledných *e* akcií používateľa) používame nasledovnú stratégiu, ktorá pre každého používateľa kombinuje vhodné globálne vzory správania P_{GL} so skupinovými vzormi správania P_{GR_i} (*i* je index zhluku používateľov do ktorého *u* patrí).

1. Ak používateľ *u* patrí do nejakej skupiny *i,* potom množina $P_{all} = P_{GL} \cup P_{GR_i}$ inak $P_{all} = P_{GL}$. Pre každý vzor správania P_j vypočítame odhadovanú podporu $\sup(P_j)$ normalizovanú na interval <0,1> (podľa veľkosti okna v prúde dát). Tiež nájdeme prienik s aktuálnym vyhodnocovacím oknom W_e pomocou algoritmu *LCS* (*longest common subset*). Označme hodnotu tohto prieniku $lcs(P_j, W_e)$ (počet spoločných akcií vzoru P_j a okna W_e, normalizovaný na hodnotu z intervalu <0,1> podľa dĺžky W_e).
2. Všetky nájdené vzory v P_{all} sa zoradia:
   a. podľa hodnoty prieniku $lcs(P_i, W_e)$ od najvyššej po najnižšiu hodnotu,
   b. podľa hodnoty podpory vzorov od najvyššej po najnižšiu hodnotu.
3. Nech *M* je mapa položiek a ich „hlasov". Iteráciou cez všetky zoradené vzory správania z P_{all} sú inkrementované „hlasy" jednotlivých položiek, ktoré sa nachádzajú vo vzoroch a nenachádzajú sa vo vyhodnocovacom okne W_e. Hodnota hlasu pre konkrétnu položku sa inkrementuje o hodnotu: $\sup(P_j) * lcs(P_j, W_e)$.
4. Nakoniec je mapa *M* zoradená zostupne podľa hodnôt hlasov jednotlivých položiek. Pre odporúčanie je vybraných najlepších *rc* (vstupný parameter metódy z ang. *recommendations count*) položiek.

6.2.3 Monitorovanie a regulácia rýchlosti spracovania sedení používateľov

Ako sme uviedli na začiatku kapitoly 4 venujúcej sa metódam dolovania vzorov správania z prúdov dát, je dôležité umožniť používateľovi nastaviť balans medzi rýchlosťou spracovania a presnosťou výsledkov. My sme sa rozhodli umožniť používateľovi nastaviť dolné obmedzenie požadovanej rýchlosti.

Naša metóda si v každom kroku udržiava informáciu o aktuálnej priemernej rýchlosti spracovania dát. Označme ju *actspeed*. Nech *ctrans* označuje aktuálny počet spracovaných transakcií od konkrétneho bodu začiatku merania. Nech *mts* označuje používateľom zadanú požadovanú minimálnu rýchlosť spracovania v transakciách za sekundu. Nech *tstart* označuje čas začiatku merania a *tend* aktuálny čas merania. Aktuálnu rýchlosť (počet spracovaných transakcií za sekundu) vypočítame ako:

$$actspeed = \frac{ctrans}{tend[s] - tstart[s]} \quad (1)$$

Čo sa týka výpočtovej náročnosti algoritmov hľadania frekventovaných množín je kritickým krokom najmä aktualizácia uzavretých frekventovaných množín s prichádzajúcou dávkou transakcií. Nech *tsupdate* označuje začiatok aktualizácie množín v *s* (od začiatku merania *tstart*). Pred každým výpočtovo náročným krokom (aktualizáciou množín) sa vypočíta maximálny povolený čas *tmax* takto:

$$tmax[s] = \frac{ctrans}{mts} - tsupdate[s] + tstart[s] \quad (2)$$

Jedná sa o hrubý zásah do výpočtu, kedy niektoré dôležité vzory nebudú nájdené. Musí teda dôjsť k určitému kompromisu, kedy znižovanie požiadavky na minimálnu rýchlosť zvyšuje presnosť algoritmu.

Zaujímavým riešením tohto problému by bolo použiť implementáciu algoritmu na dolovanie najlepších k-frekventovaných množín, kde nie je potrebné nastavovať hladinu minimálnej podpory a rýchlo sa nájde *k* najlepších vzorov (podobne ako sme opísali v časti 3.2.1). Implementácia tohto algoritmu a jeho integrácia do algoritmu *IncMine* je však už mimo rozsah tejto práce.

6.3 Sumarizácia vstupných parametrov metódy

Jedným z kritických problémov metódy je vhodné nastavenie vstupných parametrov jej individuálnych častí (zhlukovanie, hľadanie vzorov správania, odporúčanie). Pretože existuje veľké množstvo kombinácií hodnôt parametrov, môže mať ich optimalizácia exponenciálnu zložitosť. Kvôli zjednodušeniu optimalizácie je teda vhodné rozdeliť vstupné parametre do kategórií podľa ich spoločného účelu a optimalizovať jednotlivé kategórie parametrov samostatne.

Vstupné parametre sme rozdelili do 4 kategórií:

- Parametre algoritmu hľadania vzorov,
- Parametre algoritmu zhlukovania,
- Parametre odporúčania,
- Ostatné všeobecné parametre.

Parametre algoritmu hľadania vzorov:

- **Minimálna podpora** (skr. *ms* z ang. *minimal support*): Hodnota minimálnej podpory v intervale <0,1>. V zmysle výpočtu progresívnej funkcie minimálnej podpory nad posúvajúcim sa oknom uvádzanej v algoritme *IncMine* (Cheng, Ke, & Ng, 2008) je to vlastne hodnota parametra σ.
- **Miera uvoľnenia** (skr. *rr* z ang. *relaxation rate*): Je to hodnota parametra *r* v intervale <0,1> algoritmu *IncMine* (Cheng, Ke, & Ng, 2008). Určuje rýchlosť uvoľňovania potencionálne frekventovaných množín. Čím je vyššia, tým rýchlejšie sú zabúdané potencionálne frekventované množiny, ktoré nie sú frekventované, ale spĺňajú požiadavku uvoľnenej hranice minimálnej podpory (*relaxed minimum support*).
- **Dĺžka segmentu** (skr. *sl z* ang. *segment length*): Je to počet transakcií, ktoré patria do jedného segmentu. Segment je ekvivalentom časovej jednotky t_τ algoritmu

IncMine. Kvôli zjednodušeniu optimalizácie a testovania používame okno pevne fixovanej dĺžky namiesto okna s fixovaným časovým intervalom ako je uvedené aj v (Quadrana & Mestre, 2012).

- **Maximálna dĺžka množiny** (skr. *mil* z ang. *maximal itemset length*): Obmedzenie kladené na dĺžku hľadaných frekventovaných množín. Obmedzenie môže znížiť počet generovaných množín a tak zrýchliť algoritmus.
- **Veľkosť okna** (skr. *ws* z ang. *window size*): Počet posledných segmentov v ktorých sa hľadajú frekventované množiny. Zväčšovanie dĺžky okna spoločne s dĺžkou segmentov znamená pamätanie si dlhšej histórie transakcií a teda zvyšovanie výpočtovej a pamäťovej náročnosti.

Parametre algoritmu zhlukovania

- **Počet zhlukov** (skr. *gc* z ang. *groups count*): je ekvivalentný počtu hľadaných skupín používateľov.
- **Hraničný počet zmien v modeli používateľa** (skr. *tcu* z ang. *threshold number of changes in usermodel*): Počítadlo zmien v modeli používateľa *u* (*u.nsc*) je inkrementované s každým ďalším sedením používateľa. Ak toto počítadlo zmien dosiahne hodnotu vyššiu ako je hodnota parametra *tcu* tak sa počítadlo vynuluje a zmeny sa prenesú do štatistickej reprezentácie zhlukov v podobe mikrozhlukov. Parameter *tcu* zároveň určuje kapacitu radu posledných sedení v modeli používateľa *u* (atribút *u.aq*).
- **Hraničný počet zmien v mikrozhlukoch** (skr. *tcm* z ang. *threshold number of changes in microclusters*): V prípade, že došlo k dostatočnému počtu zmien v modeli používateľa tak sa tieto zmeny prenesú aj do mikrozhlukov. Počítadlo zmien v mikrozhlukoch *muc* je inkrementované s každým vykonaním tohto kroku o 1. Ak prekročí počet zmien v mikrozlukoch hodnotu danú parametrom *tcm* tak sa počítadlo *muc* vynuluje a vykoná sa makrozhlukovanie.
- **Maximálny počet mikrozhlukov** (skr. *mmc* z ang. *maximal microclusters count*): je maximálny povolený počet mikrozhlukov. Viac mikrozhlukov znamená vyššiu výpočtovú náročnosť, ale i vyššiu presnosť aproximácie dát.

Parametre odporúčania

- **Veľkosť okna vyhodnocovania** (skr. *ews* z ang. *evaluation window size*): Počet posledných akcií v sedení používateľa, na základe ktorých sa vyberajú vzory správania pre odporúčanie.
- **Počet odporúčaných položiek** (skr. *rc* z ang. *recommendations count*). Počet položiek, ktoré sú odporúčané používateľom.

Ostatné všeobecné parametre

- **Minimálna rýchlosť** (skr. *mts* z ang. *minimal transactions per second*): Minimálna požiadavka na rýchlosť (meraná ako počet transakcií za sekundu). Ak pri niektorej časovo náročnej operácií ako napr. aktualizácia množín dôjde k spomaleniu pod túto hranicu tak sa výpočet preruší (nenájdu sa všetky vzory) a pokračuje sa ďalej.
- **Hraničná hodnota rozdielu identifikátorov zhlukovania** (skr. *tcdiff z* ang. *threshold of clustering id's difference*): Hovorí aký je hraničný rozdiel medzi

počítadlom makrozhlukovania *u.lmid* v modeli používateľa *u* a počítadlom aktuálneho makrozhlukovania *lmid*. Ak platí $lmid - u.lmid > tcdiff$ tak je model používateľa označený ako neaktívny a je odstránený z pamäte.

S prípadnou modifikáciou niektorých komponentov metódy sa budú meniť aj kategórie parametrov súvisiace práve s daným účelom tej kategórie parametrov, ktorej sa to týka. Napríklad pri zmene zhlukovacieho komponentu bude pravdepodobne potrebné zmeniť niektoré parametre z tejto kategórie parametrov.

7 Realizácia metódy

V tejto kapitole opisujeme spôsob implementácie navrhnutej metódy. Podarilo sa nám implementovať experimentálny prototyp s použitím abstrakcií, ktoré sú súčasťou rámca *MOA* (Bifet, Holmes, Kirkby, & Pfahringer, 2010). Tento prototyp (7.1) sme použili na realizáciu a vyhodnotenie experimentov, ktoré opisujeme ďalej v kapitole 8. Okrem toho sme navrhli a implementovali prototyp pre aplikáciu metódy v distribuovanom prostredí pomocou technológie *Apache Storm* (7.2), ktorý rozširuje experimentálny prototyp najmä o možnosť paralelizácie úlohy dolovania vzorov správania pre jednotlivé skupiny používateľov.

7.1 Experimentálny prototyp

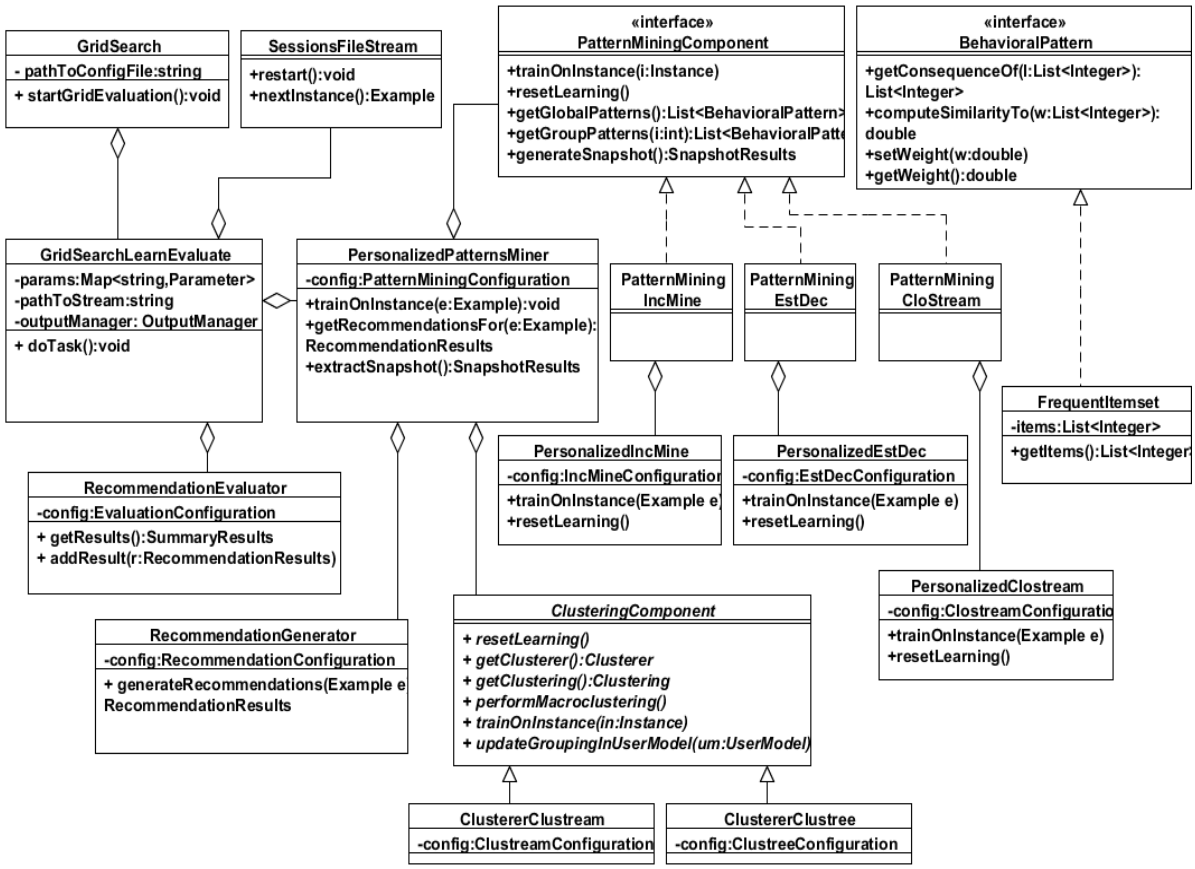
Našim cieľom bolo vytvoriť prostredie pre overenie navrhnutých konceptov a hypotéz, experimentovanie s rôznymi algoritmami zhlukovania modelov používateľa a hľadania vzorov správania v prúde dát. Na implementáciu metódy sme sa rozhodli použiť rámec *MOA* (Bifet, Holmes, Kirkby, & Pfahringer, 2010) aj kvôli dostupnosti viacerých algoritmov dolovania z prúdových dát. V tomto rámci je ako rozšírenie možné použiť implementáciu algoritmu *IncMine* tak ako ju opisujú v práci (Quadrana, Bifet, & Gavaldà, 2015). Autori implementujú tento algoritmus s niekoľkými rozdielmi oproti pôvodnej práci (Cheng, Ke, & Ng, 2008):

1. Algoritmus používa pohyblivé okno fixovanej dĺžky namiesto pohyblivého okna s fixovaným časovým intervalom, čo umožňuje jednoduchšie overenie na prúde simulovanom zo statického datasetu.
2. Na dolovanie uzavretých frekventovaných množín, čo je kritický bod celkovej výkonnosti algoritmu, použili algoritmus *CHARM* (Zaki & Hsiao, 2002) (Príloha A). Konkrétne jeho implementáciu z rámca *SFPM* (Fournier-Viger, 2016). Je to modifikovaná verzia tohto algoritmu, ktorá používa bitové množiny na reprezentáciu transakcií vo vertikálnom formáte dát.

Ďalšie algoritmy na dolovanie frekventovaných vzorov, ktoré sme sa rozhodli integrovať do prototypu pre porovnanie s nami vybraným algoritmom *IncMine* sú algoritmy *estDec+* (Shin, Lee, & Lee, 2014) a *CloStream* (Yen, Lee, Wu, & Lin, 2009) . Použili sme ich implementácie z rámca *SFPM*.

Súčasťou rámca MOA je aj implementácia algoritmu zhlukovania *CluStream* (Aggarwal C. C., Han, Wang, & Yu, 2003) s makrozhlukovacím algoritmom *k-means*, ktorý používame na zhlukovanie modelov používateľa. Pre porovnanie sme do prototypu integrovali ďalší algoritmus zhlukovania v prúde dát *ClusTree* (Kranen, Assent, Baldauf, & Seidl, 2011), ktorého implementácia je tiež dostupná v rámci MOA.

Na Obrázku 2 možno vidieť zjednodušený diagram tried znázorňujúci základnú štruktúru implementácie metódy.



Obrázok 2. Zjednodušený diagram tried experimentálneho prototypu. Sú zobrazené iba najdôležitejšie triedy a ich členovia.

Trieda *GridSearch* je vstupnou bránou metódy. Jej úlohou je načítanie konfiguračného súboru, ktorý obsahuje:

- konfiguráciu nastavenia jednotlivých experimentov (napr. vypnutie/zapnutie zhlukovania, určenie logovaných dát, ciest k vstupným a výstupným súborom),
- definuje prehľadávaný priestor hodnôt vstupných parametrov metódy.

GridSearch postupne spúšťa metódu s rôznymi konfiguráciami hodnôt parametrov.

Trieda *GridSearchLearnEvaluate* zabezpečuje beh jedného experimentu (konkrétnej konfigurácie parametrov):

- Inicializuje so správnymi hodnotami vstupných parametrov objekt *PersonalizedPatternsMiner* reprezentujúci jadro metódy a jeho súčasti:
  - komponent zhlukovania (*ClusteringComponent*),
  - komponent hľadania vzorov (*PatternMiningComponent*),
  - komponent generovania odporúčaní (*RecommendationGenerator*)
- Zabezpečuje priebeh experimentu. Teda:
  - načítanie vstupného prúdu dát (*SessionsFileStream*) a jeho presmerovanie do komponentu *PersonalizedPatternsMiner*,
  - logovanie informácií o rýchlosti spracovania,
  - extrahovanie snímok detailného stavu spracovania (vzorov, zhlukov),

- presmerovanie výsledkov odporúčania do komponentu *RecommendationEvaluator* zabezpečujúceho vyhodnotenie výsledkov odporúčania.

Trieda *PersonalizedPatternMiner* predstavuje jadro navrhovanej metódy. Pomocou komponentov zhlukovania, hľadania vzorov a generovania odporúčaní zabezpečuje vykonávanie procesu navrhnutého v časti 6.2.

Trieda *SessionsFileStream* zabezpečuje úlohu postupného načítania sedení zo vstupného dátového súboru. Trieda *OutputManager* zabezpečuje úlohy zápisu výsledkov experimentov do súborov.

Rozhranie *PatternMiningComponent* je abstrakciou pre rôzne spôsoby dolovania vzorov správania reprezentovaných rozhraním *BehavioralPattern.* My sme použili implementáciu vzorov správania vo forme frekventovaných množín akcií (*FrequentItemset*).

Implementovali sme tri rôzne spôsoby dolovania vzorov správania, kvôli možnosti porovnania nami vybraného algoritmu *IncMine* (trieda *PatternMiningIncMine*) s ďalšími dvoma algoritmami na hľadanie frekventovaných vzorov v dátach *EstDec+* (trieda *PatternMiningEstDec*) a *CloStream* (trieda *PatternMiningClostream*). Obidva algoritmy sme upravili tak aby hľadali globálne vzory správania aj skupinové vzory správania (triedy *PersonalizedIncMine, PersonalizedEstDec* a *PersonalizedClostream*).

Abstraktná trieda *ClusteringComponent* zabezpečuje základné úlohy zhlukovania používateľov do skupín a správy modelov používateľa. Túto abstrakciu sme implementovali pre nami vybraný algoritmus *CluStream*, a tiež pre algoritmus *ClusTree* s ktorým *CluStream* porovnávame.

Trieda *RecommendationGenerator* zabezpečuje samotné generovanie odporúčaní. Vstupom je aktuálne sedenie používateľa a výstupom množina odporúčaných položiek vygenerovaná na základe aktuálnych vzorov správania.

Trieda *RecommendationEvaluator* zabezpečuje vyhodnocovanie odporúčania. Počíta metriky *presnosť*, *ndcg* v jednotlivých sedeniach, pre jednotlivé skupiny používateľov aj globálne.

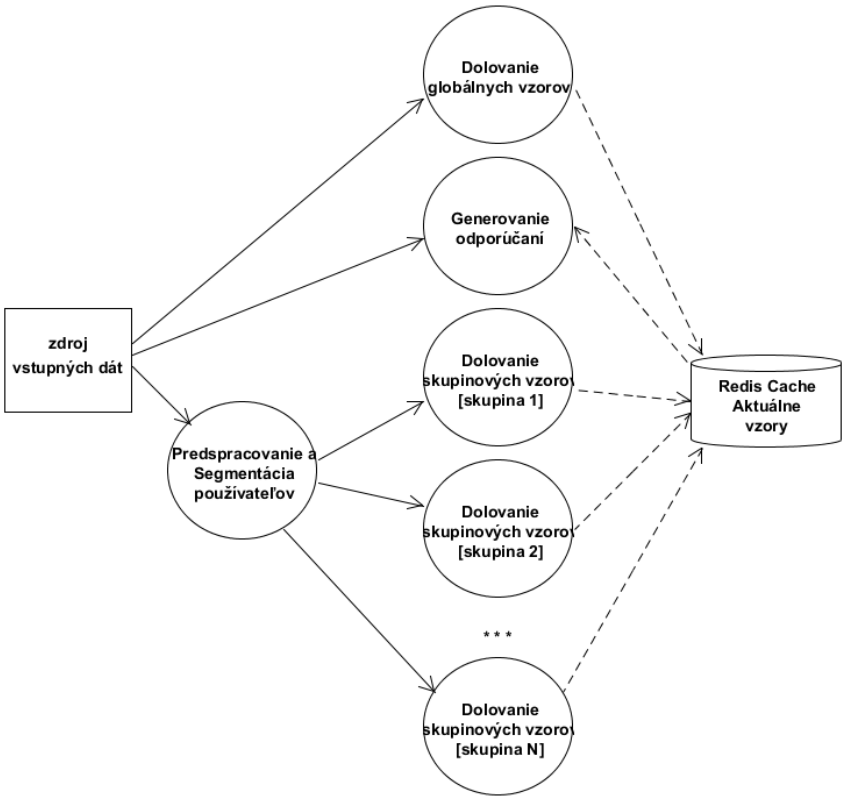
7.2 Prototyp pre distribuované prostredie

V tejto časti opisujeme spôsob implementácie metódy v distribuovanom prostredí pomocou rámca *Apache Storm*[5]. Na Obrázok 3 uvádzame základnú navrhnutú topológiu znázorňujúcu logiku realizovanej metódy vo výpočtovom klastri. Topológia je grafom pozostávajúcim z dvoch typov uzlov[6]:

[5] http://storm.apache.org

[6] Uvedené konceptuálne termíny aj notácia sú prebrané z oficiálnej dokumentácie Apache Storm dostupnej na http://storm.apache.org/

- Žriedlo (ang. *spout*) je zdroj prúdu dát. Načítava dáta z externých zdrojov a vháňa ich do topológie.
- Blesk (ang. *bolt*). V týchto uzloch sa vykonáva samotná logika spracovania dát. Transformujú prichádzajúce dáta a posielajú ich do ďalších uzlov topológie alebo externých databáz.



Obrázok 3. Navrhnutá topológia pre metódu hľadania globálnych a skupinových vzorov správania pomocou rámca Apache Storm. Uzol typu žriedlo je zobrazený ako obdĺžnik a uzol typu blesk je zobrazený ako kruh. Prerušovaná čiara znázorňuje prenos vzorov správania. Plná čiara zas prenos transakcií (sedení používateľov).

Každý tento uzol pozostáva z množiny úloh vykonávaných kdekoľvek v klastri. Každá úloha korešponduje s jedným vláknom vykonávania. V konfigurácií klastra je možné definovať počet výpočtových uzlov aj počet úloh jednotlivých výpočtových uzlov.

Nami navrhnutá topológia je veľmi jednoduchá. Uzol *dolovanie globálnych vzorov* prijíma dáta priamo zo žriedla a celkom samostatne sa stará o dolovanie globálnych vzorov správania, ktoré ukladá do úložiska *Redis*. Uzol *generovanie odporúčaní* zabezpečuje generovanie odporúčaní pre nekompletné sedenia používateľov pomocou aktuálnych vzorov správania. Uzol *predspracovanie a segmentácia používateľov* zabezpečuje predspracovanie prichádzajúcich dát o sedeniach používateľov a segmentáciu používateľov do skupín. Prichádzajúcemu sedeniu priradí označenie používateľa a skupiny do ktorej patrí. Podľa toho posiela dáta ďalej do uzla *dolovanie skupinových vzorov* pre danú skupinu. Získané vzory sa ihneď ukladajú do úložiska, odkiaľ môžu byť získané ďalšími aplikáciami.

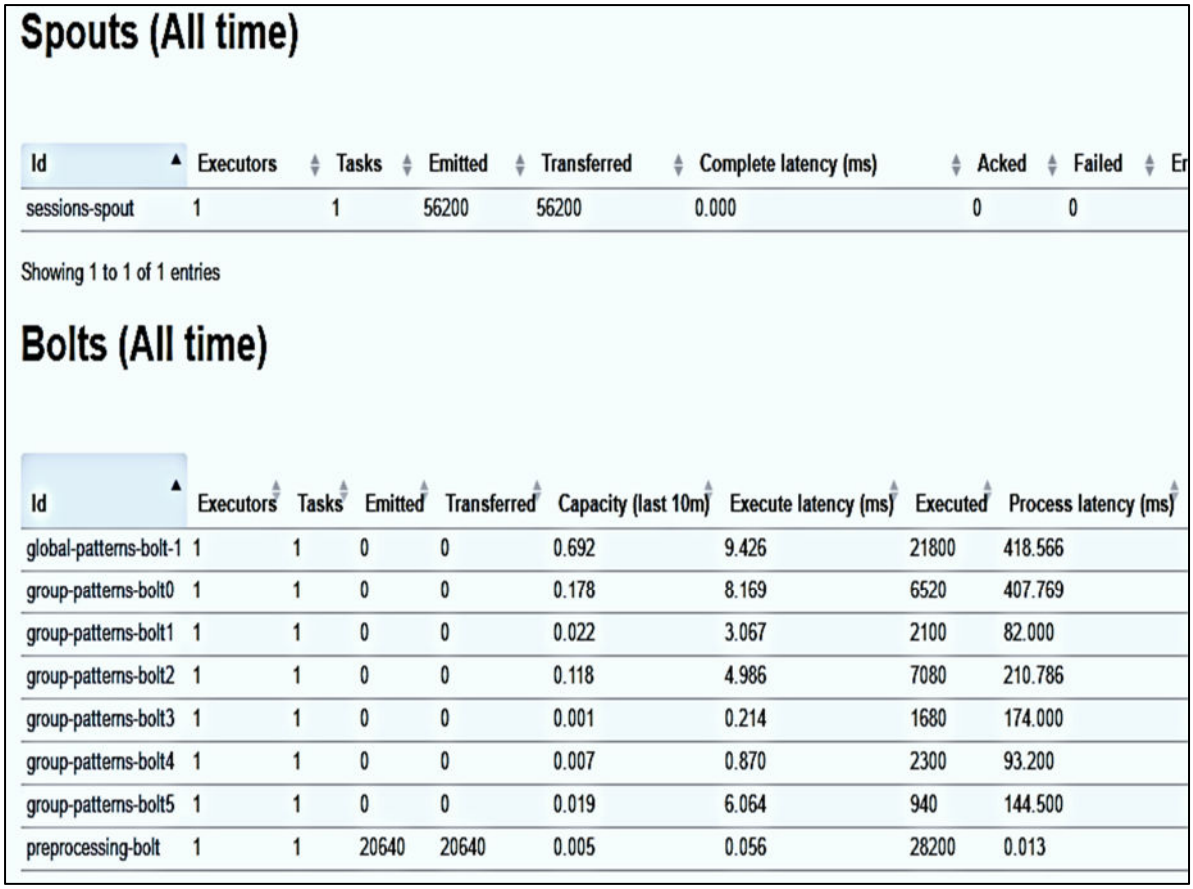
Zdroj prúdu dát je simulovaný z dátového súboru podobne ako v experimentálnom prototype. V budúcnosti by ale dáta mohli byť získavané z viacerých serverov a náš prototyp by mohol byť integrovaný so systémom spracovania správ *Apache Kafka* [7]. Ten môže byť použitý ako distribuovaný robustný rad dát, z ktorého čerpá topológia dáta a ktorý vie zvládať veľké objemy dát. Celý proces by tak pozostával z webových sídel generujúcich dáta o sedeniach používateľov do systému *Kafka* a našej topológie spracovávajúcej tieto dáta. Výsledky práce topológie (vzory správania) sú ukladané do databázy odkiaľ ich môže akákoľvek ďalšia aplikácia získať.

Topológiu sme úspešne nasadili na platforme *Azure*[8] od spoločnosti Microsoft, kde sme testovali najmä rýchlosť spracovania transakcií (8.3). Na Obrázku 4 je uvedená ilustrácia stavu topológie počas spracovania prúdu dát (počet spracovaných dát a rýchlosť spracovania v jednotlivých uzloch topológie). V uzloch *dolovanie skupinových vzorov* a *dolovanie globálnych vzorov* sa najskôr zhromaždí väčší počet transakcií (celý jeden segment (pozri parameter *sl* v 6.3), ktoré sú potom ako dávka poslané na spracovanie upravenému algoritmu *IncMine.* To je dôvod prečo je aj tzv. *Process latency,* predstavujúca čas spracovania celej dávky vyššia ako tzv. *Execute latency* predstavujúca priemerný čas spracovania jednej transakcie.

Paralelizácia jednotlivých úloh, ktoré metóda vykonáva spôsobom aký sme navrhli (ale aj iným) je možná a vedie k efektívnej kompenzácií nadbytočnej réžie, ktorú zhlukovanie a dolovanie skupinových vzorov prináša (pozri časť 8.3.3).

[7] https://kafka.apache.org/

[8] https://azure.microsoft.com/



Spouts (All time)

Id	Executors	Tasks	Emitted	Transferred	Complete latency (ms)	Acked	Failed	Er
sessions-spout	1	1	56200	56200	0.000	0	0	

Showing 1 to 1 of 1 entries

Bolts (All time)

Id	Executors	Tasks	Emitted	Transferred	Capacity (last 10m)	Execute latency (ms)	Executed	Process latency (ms)
global-patterns-bolt-1	1	1	0	0	0.692	9.426	21800	418.566
group-patterns-bolt0	1	1	0	0	0.178	8.169	6520	407.769
group-patterns-bolt1	1	1	0	0	0.022	3.067	2100	82.000
group-patterns-bolt2	1	1	0	0	0.118	4.986	7080	210.786
group-patterns-bolt3	1	1	0	0	0.001	0.214	1680	174.000
group-patterns-bolt4	1	1	0	0	0.007	0.870	2300	93.200
group-patterns-bolt5	1	1	0	0	0.019	6.064	940	144.500
preprocessing-bolt	1	1	20640	20640	0.005	0.056	28200	0.013

Obrázok 4. Ilustrácia stavu topológie počas behu spracovania dát pomocou aplikácie *Storm UI*, ktorá je súčasťou *Apache Storm*. *Executor* je nezávislé vlákno vykonávania (ang. *thread*), *Task* predstavuje konkrétnu úlohu definovanú metódou, *Emmited* predstavuje počet dátových inštancií určených na poslanie von z uzla, *Transferred* počet skutočne odoslaných inštancií. *Capacity* je percento času uzla stráveného spracovaním transakcií (zvyšok času bol uzol voľný. t.j. čakal na ďalšie dáta, hodnota blížiaca sa 1 predstavuje úzke hrdlo topológie). *Execute latency* je priemerný čas spotrebovaný na spracovanie jednej transakcie uzlom. *Executed* je počet spracovaných transakcií. *Process latency* je priemerný čas od prijatia inštancie po odoslanie tzv. *ack* potvrdenia *execute* metódou uzla.

8 Overenie metódy

Navrhovanú metódu hľadania vzorov správania sme sa rozhodli overiť nepriamo, pomocou aplikácie vzorov v úlohe odporúčania spôsobom ako sme opísali už v časti 6.2. Používateľom webového sídla odporúčame podľa aktuálnych vzorov zaujímavé stránky, resp. kategórie stránok, ktoré by mohli v ďalších krokoch navštíviť. Úspešnosť odporúčania hodnotíme prostredníctvom metriky *presnosť.* Keďže je naším cieľom spracovávať vstupné dáta čo najefektívnejšie, vyhodnocujeme tiež rýchlosť metódy ako počet spracovaných sedení používateľov za jednotku času.

Stanovili sme dve počiatočné hypotézy na ktoré sa v overení zameriavame. Prvá hypotéza je zameraná na aplikáciu vzorov správania v úlohe odporúčania. Jej znenie sme formulovali nasledovne:

H_1*: Kombináciou skupinových vzorov správania s globálnymi dokážeme zlepšiť presnosť odporúčania v porovnaní s referenčným prístupom v ktorom použijeme len globálne vzory správania alebo len skupinové vzory správania.*

Druhá hypotéza je zameraná na odhalenie prínosu skupinových vzorov správania v získavaní informácií o správaní používateľov. Jej znenie sme formulovali nasledovne:

H_2 *: Navrhovaná metóda vie odhaliť pre jednotlivé skupiny používateľov také vzory správania, ktoré sú unikátne v rámci množiny všetkých nájdených globálnych aj skupinových vzorov správania.*

Na overenie sme použili dva datasety z rozdielnych domén (e-learning a spravodajský portál) s rozličnými charakteristikami (počet používateľov, stránok, priemerná dĺžka sedení a ďalšie). Tie opisujeme bližšie v časti 8.1 spolu s opisom použitých metrík na hodnotenie úspešnosti odporúčania.

Keďže navrhovaná metóda má pomerne veľké množstvo vstupných parametrov (6.3), ako vôbec prvý krok overenia vykonávame systematické hľadanie ich najlepšej konfigurácie, maximalizujúcej presnosť a rýchlosť metódy. Kvôli efektívnosti vykonávame hľadanie samostatne pre jednotlivé časti metódy a potom spoločne pre kombinácie najlepších nastavení z týchto častí. Už pri tomto hľadaní získavame zaujímavé poznatky, ktoré opisujeme v časti 8.2.

Pri vyhodnocovaní najlepších nájdených konfigurácií parametrov porovnávame výsledky navrhovanej metódy využívajúcej kombináciu globálnych a skupinových vzorov správania voči dvom referenčným metódam. Tie sú získane úpravou navrhovanej metódy tak, že využívajú na odporúčanie iba globálne vzory správania, resp. iba skupinové vzory správania. Podrobnejšie porovnanie výsledkov týchto metód v presnosti odporúčania a rýchlosti spracovania dát uvádzame v časti 8.3. Uvádzame tiež porovnanie výsledkov metódy s použitím odlišných algoritmov zhlukovania v prúde dát (*ClusTree*) a dolovania frekventovaných množín (*EstDec+*, *CloStream*). Výsledkom tohto porovnania je, že nami navrhovaná kombinácia algoritmov *IncMine* a *CluStream* je pre použité datasety a skúmaný problém najvhodnejšia.

8.1 Použité metriky a opis dát

V tejto podkapitole opisujeme charakteristiky použitých dát a definujeme metriky (8.1.1), ktoré sú použité na vyhodnotenie úspešnosti odporúčania (8.1.2).

8.1.1 Zdroje dát

Na vyhodnotenie sme použili dva datasety z rôznych webových sídel. Prvý dataset sú dáta z webových logov adaptívneho webového výučbového systému (skr. ALEF) (Bieliková, a iní, 2014). Druhý dataset sú dáta z webových logov spravodajského portálu (skr. SP). Tieto datasety už obsahovali identifikované sedenia používateľov s použitím časovej heuristiky so štandardným nastavením 30 min (2.1).

Veľké množstvo príliš krátkych sedení pôsobí pre metódu ako šum a znižuje kvalitu nachádzaných vzorov. V datasete ALEF sme odstránili sedenia s dĺžkou 1, ktoré nevieme využiť ani na odporúčanie, keďže potrebujeme mať v každom sedení aspoň jednu akciu na nájdenie vzoru a jednu na testovanie odporúčania. V datasete SP sme odstránili aj sedenia s dĺžkou 2, ktorých bolo priveľké množstvo a pochádzali prevažne od málo aktívnych používateľov s nízkou opakovanou návštevnosťou. V Tabuľke 2 uvádzame základné charakteristiky predspracovaných datasetov.

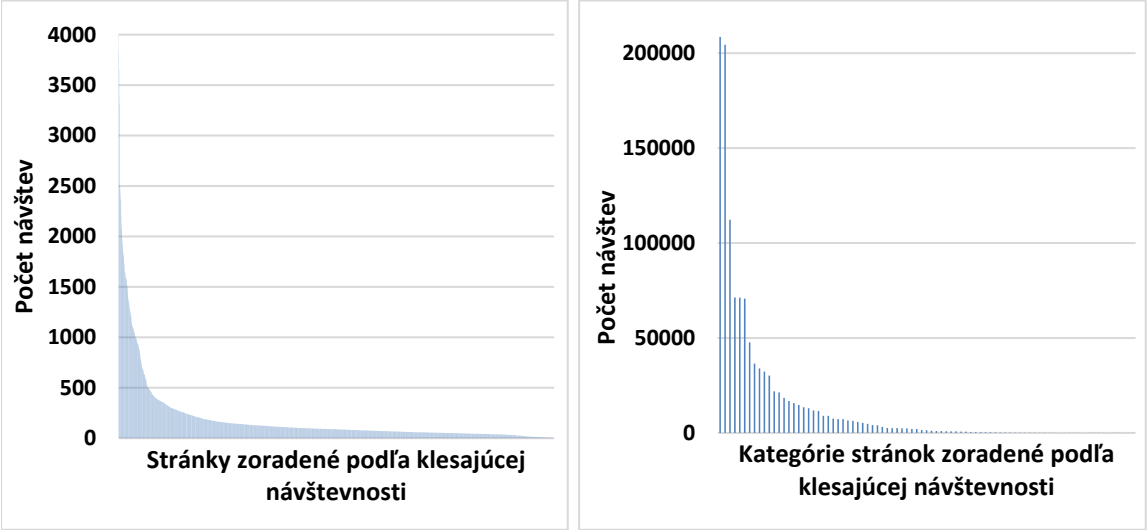
Tabuľka 2. Základné charakteristiky predspracovaných datasetov.

	E-LEARNING PORTÁL (ALEF)	SPRAVODAJSKÝ PORTÁL (SP)
počet sedení	24594	334 257
priemerná dĺžka sedenia	15.8	3.6
medián dĺžky sedenia	9.0	3.0
počet používateľov	870	199 196
časové obdobie	917 dní z toho 737 dní v ktorých bolo aspoň jedno sedenie. Kvôli tomu že ide o systém pre študentov tak aktivita používateľov logicky stúpa najmä v okolí skúškového obdobia a klesá v období prázdnin. Priemerne pripadá na jeden deň 33.37 sedení.	122 dní. Je to oveľa kratšie obdobie ako pre dataset ALEF, ale zároveň s oveľa väčším počtom akcií. Priemerne pripadá na jeden deň 2739.8 sedení.
počet možných akcií používateľa	2072 - akcia je reprezentovaná ako návšteva konkrétneho vzdelávacieho objektu používateľom (stránky).	85 - akcia je reprezentovaná ako návšteva všeobecnej kategórie obsahu (napr. šport, kultúra, lokálne, globálne správy a pod.).

Oproti datasetu ALEF je dataset SP značne väčší čo sa týka počtu sedení aj počtu používateľov. V spravodajskom portáli existujú tisíce stránok (článkov), ktoré sú aktuálne

často len krátke obdobie. Tieto stránky sme abstrahovali do 85 kategórií (napr. kultúra, šport, lokálne/globálne správy). Nad týmito kategóriami rozpoznávame vzory správania používateľov. Táto abstrakcia prináša tiež (ako je možné vidieť neskôr v časti 8.3) výrazne rýchlejšie hľadanie vzorov a kvalitu vzorov (s vyššou trvácnosťou a častým výskytom) v porovnaní s datasetom ALEF, kde hľadáme vzory správania nad konkrétnymi stránkami (vzdelávacími objektami).

Na Grafe 1 je znázornená návštevnosť stránok resp. kategórií stránok používateľmi. Z uvedených grafov je vidno, že v oboch datasetoch existuje malé množstvo položiek (stránok alebo kategórií), ktoré sú navštevované veľmi často a veľké množstvo položiek, ktoré sú navštívené len sporadicky.

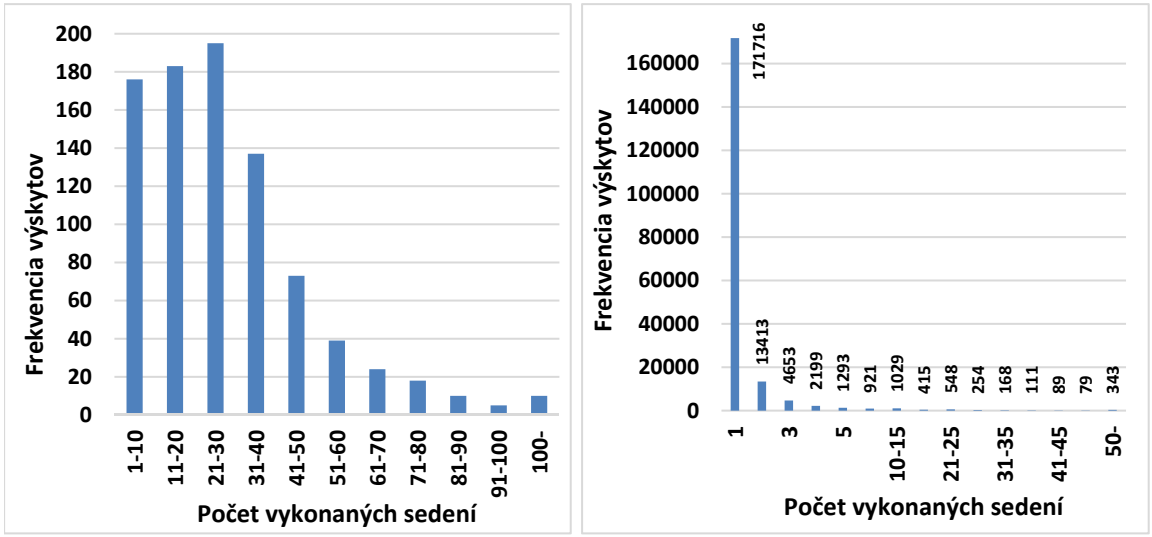


a) ALEF b) Spravodajský portál

Graf 1. Frekvencia návštev stránok používateľmi v Alefe (a) a spravodajskom portáli (b).

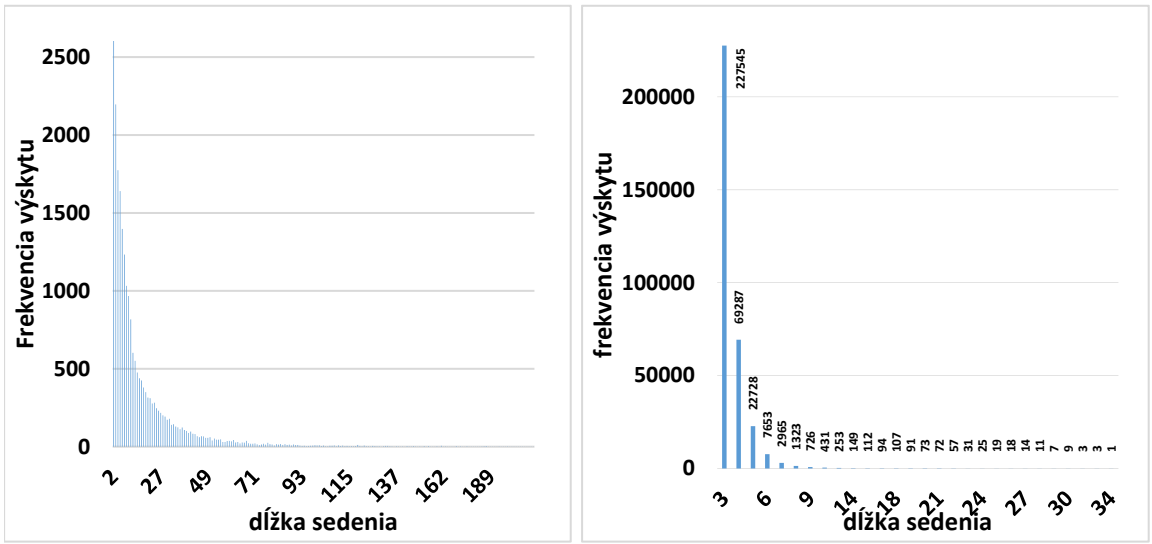
Na Grafe 2 je znázornená distribúcia počtu sedení používateľov. Z hľadiska našej metódy je vhodné aby sa používatelia vracali do webového sídla čo najviac. V tom prípade je nachádzaných tiež viac zaujímavých vzorov správania a sú aj viac využívané. Oproti datasetu ALEF je v datasete SP návratnosť používateľov omnoho nižšia. V rámci sledovaného obdobia má veľmi veľké množstvo používateľov zaznamenaných len málo sedení.

Takisto je dôležitý pohľad na distribúciu dĺžok sedení (Graf 3). Vidíme rýchle klesanie počtu sedení s ich rastúcou dĺžkou v oboch datasetoch. Sedenia v datasete SP sú tiež väčšinou omnoho kratšie ako v ALEFe. Tento fakt ovplyvňuje vyhodnocovanie. Pri zvyšovaní počtu odporúčaných položiek (parameter *rc*) a zväčšovaní veľkosti vyhodnocovacieho okna (parameter *ews*) dochádza k odfiltrovaniu väčšieho množstva sedení, ktoré nemôžu byť použité na vyhodnocovanie. Zostávajú tak čoraz dlhšie sedenia, kde intuitívne predpokladáme vyššiu úspešnosť metódy, pretože sa jedná o aktívnejších používateľov s väčším počtom vykonaných akcií.



a) ALEF b) Spravodajský portál

Graf 2. Histogram počtov vykonaných sedení používateľov.



a) ALEF b) Spravodajský portál

Graf 3. Histogram dĺžok sedení. Predspracovaný dataset ALEF (a) s odstránením krátkych sedení (1 položka). Predspracovaný dataset SP (b) s odstránením krátkych sedení (1-2 položky).

8.1.2 Spôsob overenia úspešnosti odporúčania

Úspešnosť odporúčania hodnotíme pomocou metriky *presnosť* (ang. *precision*). Konkrétnejšie sme túto metriku zisťovali pre počty odporúčaných položiek 1, 2, 3, 4, 5, 10, 15, keďže v zvolených doménach je väčšina sedení pomerne krátka. Každé prichádzajúce používateľské sedenie S je reprezentované vektorom akcií $S = < a_1, a_2, \dots a_n >$. Sedenie S sa rozdelí na 2 časti:

- trénovaciu množinu $A = < a_1, a_2, \ldots, a_{ews} >$, ktorá sa použije na hľadanie najvýstižnejších vzorov správania podobných k aktuálnemu sedeniu (6.2) a generovanie odporúčania,
- a testovaciu množinu $B = < a_{ews+1}, \ldots, a_n >$, ktorá sa použije vyhodnotenie odporúčania.

Vstupným parametrom metódy je veľkosť okna vyhodnocovania *ews* (skr. z ang. *evaluation window size*). Nech množina odporúčaných položiek je R. Ak aktuálne sedenie S nespĺňa podmienku $|S| \geq (ews + |R|)$ tak ho pre účely vyhodnocovania odporúčania ignorujeme. *Presnosť* vypočítame ako:

$$precision = \frac{|R \cap B|}{|R|} \tag{3}$$

8.2 Optimalizácia

V tejto časti vyhodnocujeme úspešnosť odporúčania pre rôzne konfigurácie vstupných parametrov metódy. V časti 8.2.1 opisujeme metodológiu systematického hľadania najlepšej konfigurácie vstupných parametrov. V častiach 8.2.2 až 8.2.5 uvádzame výsledky hodnotenia úspešnosti odporúčania rôznych konfigurácií vstupných parametrov v rámci jednotlivých kategórií parametrov. V časti 8.2.6 uvádzame záverečné vyhodnotenie a výber celkovo najlepších konfigurácií. V prílohe F uvádzame dodatočné materiály k vyhodnoteniu (grafy, tabuľky). Detailné výsledky uvádzame v prílohe na elektronickom médiu.

8.2.1 Metodológia optimalizácie

Navrhovaná metóda kombinuje dve existujúce metódy dolovania z prúdu dát, ktoré používajú viacero parametrov a sama vyžaduje od používateľa definovať niekoľko hodnôt (6.3). Preto sme sa rozhodli venovať podstatnú časť vyhodnotenia skúmaniu vplyvov zmien hodnôt týchto parametrov na úspešnosť v úlohe odporúčania.

Testovaním rôznych nastavení parametrov hľadáme ich najlepšiu konfiguráciu z hľadiska *presnosti* odporúčania. Aby sme zmenšili počet rôznych skúmaných konfigurácií tak sme parametre rozdelili do 4 kategórií podľa ich vzájomného súvisu tak ako sme uviedli v časti 6.3. V rámci každej kategórie skúmame vplyv zmien konkrétnych hodnôt parametrov na výsledky odporúčania. Z každej kategórie vyberáme konfigurácie, ktoré sú najsľubnejšie čo sa týka *presnosti* a zároveň rýchlosti.

Po hľadaní najlepších konfigurácií v rámci jednej kategórie parametrov berieme ako vstupné nastavenie do testovania ďalšej kategórie (pre ostatné parametre metódy) dovtedy najlepšie nastavenie z hľadiska presnosti odporúčania a rýchlosti.

V závere testujeme konfigurácie parametrov, ktoré vznikli skombinovaním vybraných najlepších nastavení z jednotlivých kategórií. Zo získaných výsledkov vyberáme celkovo najlepšiu konfiguráciu.

Počiatočné nastavenie parametrov a voľbu testovaných hodnôt parametrov v jednotlivých skupinách sme získali iniciálnym odhadom experta. V Tabuľke 3 uvádzame rozdelenie parametrov do skupín a testované nastavenia parametrov.

Tabuľka 3. Testované nastavenia parametrov pri hľadaní najlepšej konfigurácie.

parameter	kategória	testované hodnoty
ms	Dolovanie vzorov	0.005, 0.01, 0.02, 0.03 0.04, 0.05, 0.1
rr	Dolovanie vzorov	0.1, 0.5, 0.9
sl	Dolovanie vzorov	25, 50, 100, 150, 200, 500
ws	Dolovanie vzorov	5,10,15
gc	Zhlukovanie	2, 4, 6, 8
tcu	Zhlukovanie	5,10,15
tcm	Zhlukovanie	50, 100, 200, 400, 800
mmc	Zhlukovanie	100,1000
ews	Odporúčanie	1,2,3,4,5,6,7, 8, 9, 10
rc	Odporúčanie	1,2,3,4,5,10,15
mts	Všeobecné	15
tcdiff	Všeobecné	1 - 20

Nech h je maximálny počet aktuálnych sedení, ktoré sú metódou udržované v pamäti. Vypočíta sa ako: $h = ws * sl$ (veľkosť okna a dĺžka segmentu). Vyhodnocovanie každej konfigurácie začína až po počte spracovaných sedení, vypočítanom ako: $h_m = \max_{c \in C} h_c$. Kde C je množina všetkých prehľadávaných konfigurácií parametrov. Eliminujeme tak rozdiely jednotlivých konfigurácií na začiatku spracovania prúdu dát, kedy konfigurácie s menšou dĺžkou h dokážu odporúčať skôr ako konfigurácie s väčšou dĺžkou h.

Ďalším obmedzením v našom testovaní je obmedzenie rýchlosti (parameter *mts*). Rozhodli sme sa pre každú testovanú konfiguráciu fixovať minimálnu povolenú hodnotu rýchlosti na 15 transakcií/sekunda v prípade ALEFu a 150 transakcií/sekunda v prípade SP.

8.2.2 Optimalizácia parametrov algoritmu dolovania vzorov

V tejto časti sa venujeme vyhodnoteniu úspešnosti odporúčania s rôznymi konfiguráciami kategórie parametrov pre algoritmus dolovania uzavretých frekventovaných množín v prúde dát *IncMine* (272 rôznych konfigurácií).

Na základe *presnosti* odporúčania sme vybrali najlepšie konfigurácie uvedené v Tabuľke 4. Sledovali sme vplyv jednotlivých parametrov a kombinácií parametrov na výsledky odporúčania. Zistili sme, že konfigurácie s kratšou dĺžkou segmentov (*sl, 25-50*), dlhším oknom (*ws, 10-15*) a nižšou mierou uvoľňovania (*rr, 0.1-0.5)* dosahovali priemerne lepšie výsledky *presnosti*. To znamená, že častejšie dochádza k aktualizácií vzorov (po každom spracovanom segmente), pri zachovaní pomalšieho (opatrnejšieho) uvoľňovania potencionálne frekventovaných vzorov (vďaka vyššej *ws* a nižšej *rr*).

Ďalej sme sledovali, že príliš nízke hodnoty *ms* spôsobujú generovanie veľkého množstva vzorov, čo značne spomaľuje metódu (v prípade ALEFu až na spodnú hranicu obmedzenia rýchlosti). Na druhej strane, príliš vysoké hodnoty *ms* generujú malé množstvo vzorov, čo môže spôsobiť ignorovanie niektorých potencionálne dôležitých vzorov.

Tabuľka 4. Najlepšie konfigurácie parametrov algoritmu dolovania vzorov. Nameraná *rýchlosť* v transakciách za sekundu, *presnosť* pre rôzne počty odporúčaných položiek (výsledky sú uvádzané v %). Sú to konfigurácie, ktoré sa umiestnili na prvých dvoch miestach v rámci metriky *presnosť* pre rôzne počty odporúčaných položiek. Intenzita farby odráža mieru úspešnosti v rámci hodnôt v danom stĺpci.

	ms	rr	sl	mil	ws	rýchlosť (t/s)	P@1	P@2	P@3	P@4	P@5	P@10	P@15
E-LEARNING PORTÁL (ALEF)													
1	0.05	0.1	25	10	15	61.5	48.03	48.17	48.00	47.86	47.47	46.49	44.32
2	0.04	0.5	50	10	10	15.4	47.72	48.76	48.55	48.29	48.17	47.25	45.76
3	0.03	0.5	50	10	10	15.5	47.93	48.34	48.03	47.82	47.69	46.82	45.27
4	0.05	0.5	25	10	15	15.4	47.89	48.73	48.44	48.29	48.07	47.07	45.69
5	0.04	0.1	50	10	10	15.5	47.76	48.31	48.45	48.17	48.17	47.08	45.56
SPRAVODAJSKÝ PORTÁL													
1	0.05	0.5	25	10	15	3892.3	64.33	53.20	55.20	55.07	54.18	49.88	41.77
2	0.03	0.1	50	10	15	3065.1	64.04	53.25	55.46	55.84	55.78	57.15	50.88
3	0.005	0.1	500	10	15	574.2	52.84	45.27	49.46	52.54	56.33	69.67	72.06
4	0.05	0.1	25	10	15	3623.3	64.29	53.23	55.29	55.24	54.48	51.37	43.88
5	0.03	0.1	50	10	10	3812.2	63.88	53.14	55.38	55.56	55.17	53.83	46.69
6	0.04	0.1	50	10	15	3409.1	63.99	53.11	55.32	55.62	55.36	55.58	48.23
7	0.01	0.1	150	10	15	1688.4	60.92	52.00	54.07	55.08	56.32	64.49	64.40
8	0.005	0.5	500	10	15	999.3	57.20	49.88	51.70	53.02	56.09	68.95	69.82

Čo sa týka rýchlosti spracovania transakcií je vidno značný rozdiel v rýchlosti medzi datasetmi. Už sme uviedli v časti venujúcej sa opisu dát (8.1.1), že je to spôsobené rozdielom v dimenzionalite akcií nad ktorými sa hľadajú vzory správania.

V datasete ALEF všetky vybrané konfigurácie okrem konfigurácie 1 spadli až na spodnú hranicu minimálnej požadovanej rýchlosti (15 transakcií/sekunda). Konfigurácia 1 má oproti konfigurácií 2 o polovicu kratšiu *sl* a o *0.01* vyššiu *ms*. Je zaujímavé, že takýto rozdiel spôsobí významnú zmenu čo sa týka rýchlosti spracovania aj presnosti odporúčania. Ak teda je dôležitá rýchlosť je dobrou voľbou konfigurácia 1. Ak nie je rýchlosť až tak kritická a dôležitá je presnosť pri odporúčaní 2 a viac položiek tak je lepšou voľbou konfigurácia 2.

V datasete SP sú najzaujímavejšie konfigurácie parametrov dosahujúce vysokú rýchlosť a tiež presnosť, konfigurácie 1 a 2.

Do záverečného vyhodnocovania sme teda vybrali hodnoty parametrov práve z konfigurácií 1 a 2 z datasetu ALEF a z konfigurácií 1 a 2 z datasetu SP.

8.2.3 Optimalizácia parametrov algoritmu zhlukovania

V tejto časti sa venujeme vyhodnoteniu úspešnosti odporúčania s rôznymi konfiguráciami kategórie parametrov zhlukovania (120 rôznych konfigurácií). Vybrali sme najlepšie konfigurácie uvedené v Tabuľke 5.

Sledujeme, že zhlukovanie používateľov do menšieho počtu skupín (< 6) je menej úspešne v oboch datasetoch. Pravdepodobne z dôvodu, že sú v tom prípade niektoré rozdielne skupiny používateľov nesprávne spájané.

V oboch doménach sú najlepšie výsledky dosahované s použitím nižšej hodnoty parametra *tuc* (5) v kombinácií s vyššími hodnotami *tcm* (400-800). To znamená, že mikrozhluky sú častejšie aktualizované s čo najaktuálnejším správaním používateľov.

A tiež, že počet zmien v mikrozhlukoch, potrebný na vykonanie makrozhlukovania, je dostatočne veľký aby sa zachytilo potrebné množstvo informácií na vytvorenie správnych zhlukov používateľov.

Tabuľka 5. Najlepšie konfigurácie parametrov zhlukovania. Nameraná rýchlosť v transakciách za sekundu, *presnosť* pre rôzne počty odporúčaných položiek (výsledky sú uvádzané v %). Sú to konfigurácie, ktoré sa umiestnili na prvých dvoch miestach v rámci metriky *presnosť* pre rôzne počty odporúčaných položiek. Intenzita farby odráža mieru úspešnosti v rámci hodnôt v stĺpci.

	gc	tuc	tcm	mmc	rýchlosť (t/s)	P@1	P@2	P@3	P@4	P@5	P@10	P@15
E-LEARNING PORTÁL (ALEF)												
1	6	5	400	100	50.7	48.78	48.78	48.60	48.16	47.94	46.96	45.14
2	8	5	400	100	52.7	48.70	48.65	48.59	48.37	48.05	46.85	44.89
3	8	5	800	100	58.4	48.42	48.66	48.40	48.00	47.79	46.72	44.77
4	4	5	400	100	47.5	48.57	48.53	48.27	47.99	47.77	46.59	44.61
5	4	5	200	100	54.0	48.57	48.57	48.29	48.19	47.82	46.76	44.78
6	8	5	200	100	54.0	48.52	48.52	48.47	48.19	47.96	46.87	44.86
7	6	5	200	100	47.7	48.27	48.39	48.35	48.11	47.88	46.90	44.90
8	4	5	200	1000	56.9	48.22	48.54	48.33	48.11	47.86	46.70	44.92
SPRAVODAJSKÝ PORTÁL												
1	8	5	800	1000	1963.1	64.55	53.77	55.93	56.37	56.33	57.96	51.18
2	8	5	400	1000	1994.8	64.55	53.69	55.89	56.31	56.32	57.65	51.28
3	6	5	400	100	2869.3	64.29	53.42	55.52	55.89	56.00	57.69	51.37
4	6	5	800	1000	2018.3	64.48	53.75	55.84	56.19	56.23	57.92	51.24

V prípade datasetu SP sledujeme vyššiu úspešnosť konfigurácií s vyššou hodnotou *mmc* (1000). V prípade ALEFu sú úspešnejšie konfigurácie s nižšou hodnotou *mmc* (100).

Čo sa týka rozdielov v rýchlostiach, tie už nie sú také značné ako pri rôznych konfiguráciách z kategórie parametrov algoritmu dolovania vzorov.

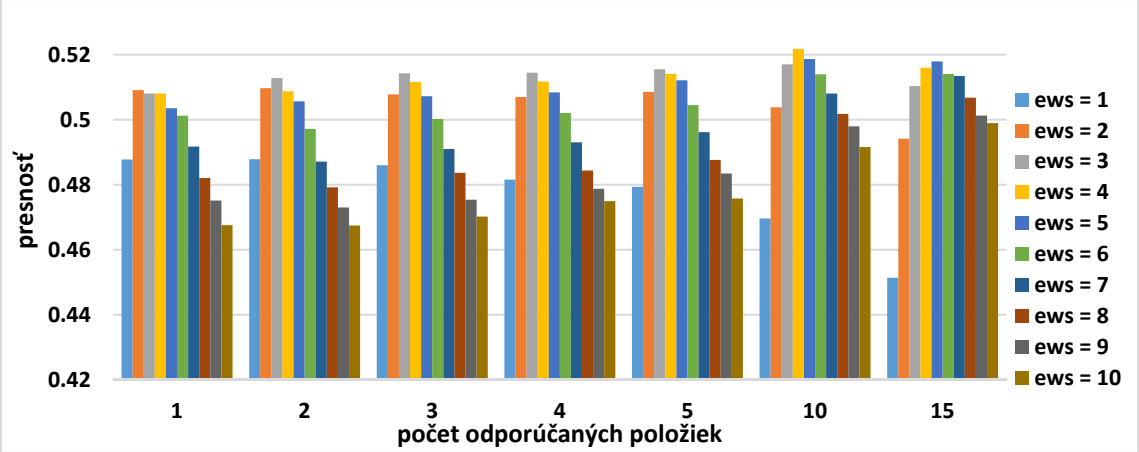
Do záverečného vyhodnocovania sme teda vybrali hodnoty parametrov z konfigurácií 1, 2 (kvôli dobrej presnosti) a 3 (kvôli vyššej rýchlosti) z datasetu ALEF a z konfigurácií 1, 2 (kvôli dobrej presnosti) a 3 (kvôli vyššej rýchlosti) z datasetu SP.

8.2.4 Optimalizácia parametrov odporúčania

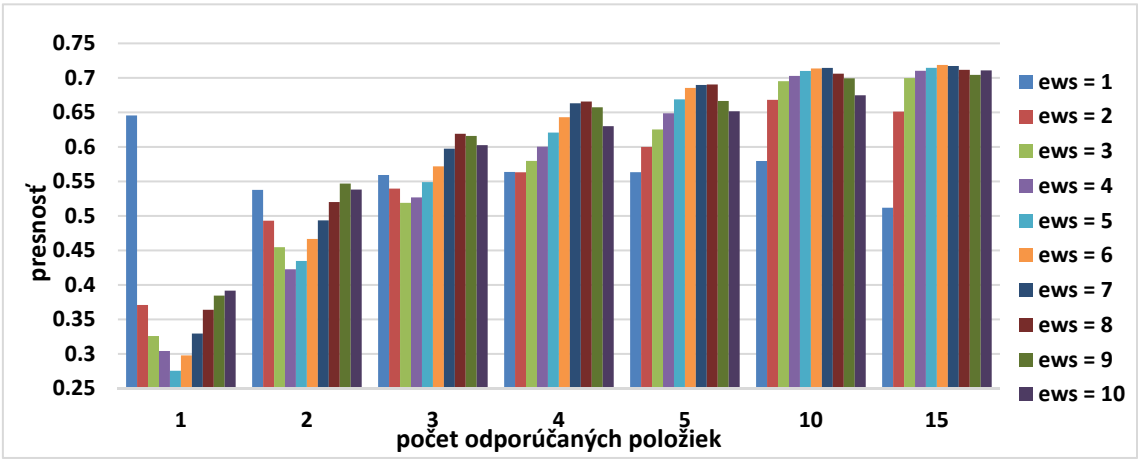
V tejto časti sa venujeme vyhodnoteniu úspešnosti odporúčania s rôznymi konfiguráciami kategórie parametrov týkajúcich sa úlohy odporúčania. V tejto kategórií pozorujeme len parameter *ews* (veľkosť vyhodnocovacieho okna) hoci tu patrí aj parameter *rc* (počet odporúčaných položiek), ktorého rôzne hodnoty sledujeme implicitne pri každej konfigurácií aj v ostatných kategóriách parametrov.

Zmeny hodnôt parametrov *ews* a *rc* menia tiež počet sedení, ktoré môžu byť použité na vyhodnocovanie. Nemôžme teda priamo porovnávať konfigurácie s rozdielnymi hodnotami *ews* a *rc* (bolo by to možné iba ak by sme ignorovali veľmi veľký počet krátkych sedení). Napriek tomu uvádzame pozorovania výsledkov s rôznymi hodnotami *ews* a *rc*, ktoré ozrejmujú význam týchto parametrov.

Sledujeme, že v datasete ALEF sú lepšie výsledky *presnosti* dosahované pre *ews* >= *2* a *ews* <= *6* (Graf 4). V datasete SP s veľkým množstvom krátkych sedení, konfigurácie s *ews* = *1* prinášajú výnimočne dobré výsledky pri odporúčaní len jednej položky. Pri odporúčaní viacerých položiek dosahujú lepšie výsledky s vyšším *ews* (Graf 5).



Graf 4. Výsledky metriky *presnosť* pre rôzne veľkosti vyhodnocovacieho okna *ews* pre rôzne počty odporúčaných položiek v datasete ALEF.



Graf 5. Výsledky metriky *presnosť* pre rôzne veľkosti vyhodnocovacieho okna *ews* pre rôzne počty odporúčaných položiek v datasete SP.

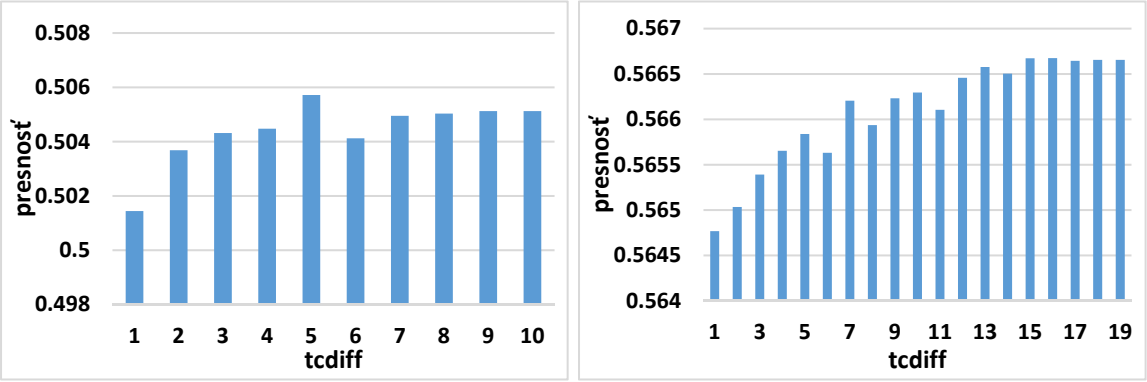
Krátke vyhodnocovacie okno dáva priestor možnosti výberu veľkého množstva vzorov správania (aj kratších aj dlhších), ktoré majú s ním prienik a môžu byť zapojené do odporúčania. To je však zároveň aj obmedzením, pretože okno je príliš malé na to aby rozpoznalo charakter sedenia a zámer používateľa. Pri väčších oknách sa *presnosť* opäť znižuje, čo je zrejme spôsobené aj tým, že neexistuje dostatok dlhších vzorov, ktoré by vedeli dané správanie používateľa správne charakterizovať.

Tento parameter je teda špecifický a do výsledného hľadania najlepšej konfigurácie (8.2.6) sme vybrali hodnoty *ews* z množiny {2, 3, 4} pre ALEF a {1,2,3,4,5} pre SP, pri ktorých nebola množina vyhodnocovaných sedení priveľmi zmenšená a zároveň boli dosiahnuté najlepšie výsledky presnosti.

8.2.5 Optimalizácia ostatných parametrov

V tejto časti sa venujeme vyhodnoteniu odporúčania v spojitosti s rôznymi konfiguráciami kategórie ostatných parametrov metódy (*všeobecné*). Sledujeme rôzne hodnoty parametra *tcdiff*. Rôzne hodnoty parametra pre obmedzenie minimálnej rýchlosti (*mts*) testujeme až vo vyhodnotení metódy (8.3).

Staré modely používateľov by mali byť odstraňované aby sa tak zabránilo potencionálnemu zaplneniu pamäte a spomaľovaniu spracovania po dlhšej dobe prevádzky. Odstraňovanie neaktívnych používateľov príliš skoro (keď je *tcdiff* nízke) vždy po novom uskutočnenom makrozhlukovaní vykazuje mierne horšie výsledky *presnosti.* V oboch datasetoch sledujeme trend mierneho zlepšovania výsledkov *presnosti* so stúpajúcou hodnotou parametra *tcdiff* (Graf 6). Celkovo k makrozhlukovaniu dochádza v ALEFe 9 krát a v SP 18 krát. Preto testujeme rôzne hodnoty *tcdiff* po túto hodnotu.



a) ALEF

b) Spravodajský portál

Graf 6. Mierne rastúci charakter priemernej presnosti (pre rôzne počty odporúčaných položiek) so zmenou hodnoty parametra *tcdiff* v datasete ALEF (a) a Spravodajský portál(b).

Sledovaný trend nasvedčuje tomu, že je dobré snažiť sa o ponechanie modelov používateľa čo najdlhšiu dobu až pokiaľ to dovolia požiadavky na pamäťovú náročnosť.

Do záverečného vyhodnotenia sme vybrali hodnoty parametra *tcdiff* = 5 pre dataset ALEF a *tcdiff=15* pre SP, pri ktorých sa sledovaný trend pozastavil.

8.2.6 Výber optimálnej konfigurácie

Na základe vybraných najlepších konfigurácií zo všetkých kategórií parametrov ako ich uvádzame v predchádzajúcich častiach sme skonštruovali priestor prehľadávania v ktorom hľadáme celkovo najlepšiu konfiguráciu. Najlepšie vybrané konfigurácie sú zobrazené v Tabuľke 6. Výsledky sme sledovali samostatne pre rôzne hodnoty parametra *ews,* ktorý, ako sme už spomínali, mení veľkosť priestoru vyhodnocovania.

Z datasetu ALEF sme vybrali ako najlepšiu konfiguráciu 1, ktorá dosahuje dobré výsledky presnosti a zároveň je najrýchlejšia z vybraných. Z datasetu SP sme vybrali konfiguráciu 1, ktorá vyniká aj rýchlosťou aj presnosťou najmä pri menšom počte odporúčaných položiek (veľmi podobné výsledky dosahuje aj konfigurácia 2).

Tabuľka 6. Najlepšie konfigurácie parametrov. Nameraná rýchlosť v transakciách za sekundu, *presnosť* spriemerovaná pre rôzne *ews* (výsledky sú uvádzané v %). Sú to konfigurácie, ktoré sa umiestnili na prvých dvoch miestach v rámci metriky presnosť pre rôzne počty odporúčaných položiek. Intenzita farby odráža mieru úspešnosti v rámci hodnôt v stĺpci.

	ms	rr	sl	mil	ws	gc	tuc	tcm	mmc	tcdiff	rýchlosť (t/s)	P@1	P@2	P@3	P@4	P@5	P@10	P@15
E-LEARNING PORTÁL (ALEF)																		
1	0.05	0	25	10	15	6	5	400	100	5	49.4	50.84	51.04	51.12	51.10	51.27	51.42	50.68
2	0.04	0	50	10	15	8	5	800	100	5	15.4	49.95	50.64	51.09	51.16	51.32	51.59	51.06
3	0.05	0	50	10	15	8	5	400	100	5	15.4	50.15	50.70	50.96	51.11	51.35	51.62	51.13
4	0.04	0	50	10	10	6	5	400	100	5	15.5	49.95	50.62	51.12	51.24	51.48	51.83	51.32
5	0.05	0	50	10	15	6	5	800	100	5	15.3	50.11	50.71	51.15	51.29	51.51	51.66	51.14
6	0.04	0	50	10	10	8	5	800	100	5	15.5	49.78	50.52	50.82	50.95	51.25	51.61	51.12
7	0.05	0	50	10	15	6	5	400	100	5	15.4	50.01	50.80	51.15	51.25	51.36	51.52	50.99
SPRAVODAJSKÝ PORTÁL																		
1	0.05	1	25	10	15	8	5	800	1000	15	2311.2	39.10	47.36	53.98	58.19	61.33	64.40	61.42
2	0.05	1	25	10	15	8	5	400	1000	15	2348.9	39.25	47.35	53.88	57.99	61.13	64.24	61.51
3	0.05	0	25	10	15	8	5	800	1000	15	2246.8	39.08	47.35	54.01	58.19	61.41	64.59	62.18
4	0.03	0	50	10	15	8	5	800	1000	15	2092.3	38.59	46.92	53.95	58.60	62.27	67.16	65.72
5	0.01	0	500	10	15	8	5	800	1000	15	597.1	33.14	42.73	51.40	57.95	63.22	73.42	74.75
6	0.01	0	500	10	15	6	5	400	100	15	626.8	32.56	42.12	50.87	57.48	62.70	73.18	74.79
7	0.03	1	50	10	15	8	5	800	1000	15	2302.0	38.61	47.03	54.04	58.49	62.05	66.75	64.59
8	0.05	0	25	10	15	8	5	400	1000	15	2262.5	39.24	47.32	53.91	58.04	61.22	64.44	62.26
9	0.01	1	500	10	15	6	5	800	1000	15	1051.5	33.84	43.43	51.56	57.90	63.00	73.25	74.74
10	0.01	1	500	10	15	8	5	800	100	15	1146.8	33.19	42.75	50.94	57.31	62.56	73.18	74.72
11	0.05	0	50	10	15	8	5	800	1000	15	2328.6	38.80	47.10	54.01	58.53	62.12	66.23	64.26
12	0.01	1	500	10	15	8	5	800	1000	15	1013.0	33.88	43.52	51.70	57.97	63.08	73.40	74.77
13	0.01	0	500	10	15	6	5	800	1000	15	605.8	32.69	42.28	51.06	57.80	63.19	73.28	74.74

8.3 Vyhodnotenie navrhnutej metódy

V tejto podkapitole pre najlepšie konfigurácie parametrov objavené v predchádzajúcej časti podrobnejšie vyhodnocujeme vzťahy medzi výsledkami (presnosť, rýchlosť) navrhovanej metódy využívajúcej kombináciu globálnych a skupinových vzorov správania (skr. *GG*) voči referenčným metódam využívajúcim buď len globálne vzory správania (skr. *GL*), alebo len skupinové vzory správania (skr. *GR*). V časti 8.3.1 uvádzame spôsob akým vyhodnocujeme prínos kombinácie skupinových vzorov správania s globálnymi v navrhovanej metóde a ako porovnávame navrhovanú metódu (GG) voči referenčným (*GL* a *GR*). Následne v časti 8.3.2 uvádzame výsledky štatistického testu, ktorý súhlasí s hypotézou H_1 stanovenou v úvode kapitoly a výsledky porovnania metód, ktoré súhlasia s hypotézou H_2. V časti 8.3.3 uvedené metódy porovnávame z hľadiska rýchlosti spracovania dát a v časti 8.3.4 sa porovnávame s niektorými ďalšími variantami algoritmov zhlukovania a hľadania frekventovaných vzorov.

8.3.1 Metodológia vyhodnotenia úspešnosti

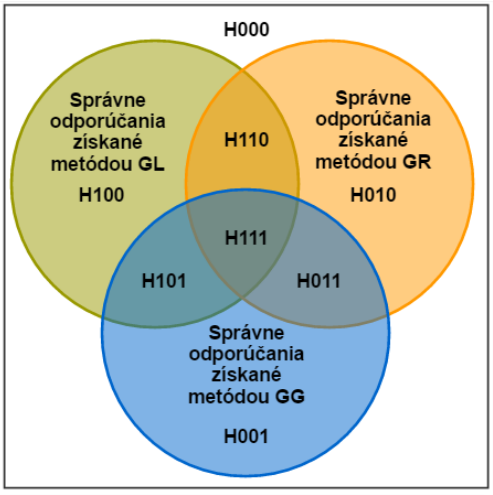
V priebehu testovania metódy generujeme v čase niekoľko snímok aktuálneho stavu nájdených vzorov. Pre každú kategóriu vzorov (globálne a skupinové) sledujeme:

- počty unikátnych vzorov, ktoré sme vedeli získať pomocou metódy,
- počet používateľov v skupine,
- pomer priemernej podpory (frekvencie výskytov) unikátnych vzorov k priemernej podpore všetkých vzorov.

Overujeme tak či metóda dokáže nájsť unikátne vzory správania pre jednotlivé skupiny používateľov tak ako to hovorí hypotéza H_2.

Okrem toho sledujeme počas behu experimentu pre každé používateľské sedenie počet správne odporúčaných položiek pre jednotlivé metódy odporúčania podľa toho aké vzory používajú. Sledujeme tri samostatné množiny tak ako sú znázornené na vennovom diagrame (Graf 7). Ide o množiny všetkých správne odporúčaných položiek v sedeniach používateľov získaných pomocou metódy:

- využívajúcej len globálne vzory (ozn. *GL*) (1. množina),
- využívajúcej len skupinové vzory (ozn. *GR*) (2. množina),
- využívajúcej kombináciu skupinových a globálnych vzorov (ozn. *GG*) (3. množina).



Graf 7. Vennov diagram znázorňuje označenia jednotlivých množín vektorov úspešnosti odporúčania a ich prienikov.

Uvedieme príklad pre jedno sedenie používateľa. Nech je počet odporúčaných položiek 5. Vektory správnosti odporúčania pre jednotlivé metódy odporúčania nech sú pre konkrétne sedenie používateľa nasledovné (hodnota *1* na i-tej pozícií znamená, že i-ta odporúčaná položka bola správna a *0*, že odporúčaná položka bola nesprávna):

- Vektor úspešnosti odporúčania získaný pomocou metódy *GL*: **11001**,
- získaný pomocou metódy *GR*: **01100**,
- získaný pomocou metódy *GG*: **11101**.

Rôzne prieniky týchto vektorov označujeme ďalej ako písmeno *H* (z ang. *hits*) nasledované troma bitmi kde prvý bit označuje metódu *GL*, druhý bit označuje metódu *GR*, tretí bit označuje metódu *GG*. Teda napr. *H101* predstavuje počet úspešných odporúčaní získaných pomocou *GL* a zároveň aj pomocou *GG* ale nie pomocou *GR*. Teda pre uvedený príklad by to bol prienik vektorov 11001 (*GL*) a 11101 (*GG*) a 10011 (negované *GR*) čo je 10001.

Hodnota *H101* je teda suma výsledného vektora a to je 2. Pre jednoduchšie pochopenie sme znázornili uvedené množiny a ich prieniky pomocou vennovho diagramu (Graf 7).

Pomocou sumarizácie množín vektorov úspešnosti odporúčaní a ich prienikov pre všetky sedenia používateľov vieme zistiť zaujímavé fakty ako napr.:

- Počet správnych odporúčaní len pomocou *GR* a nie *GL*: *H011+H010.*
- Počet správnych odporúčaní len pomocou *GL* a nie *GR*: *H101+H100.*
- Počet správnych odporúčaní len pomocou *GG* a nie iných: *H001.*
- Počet správnych odporúčaní len pomocou *GR*, ktoré sme kombináciou skupinových a globálnych vzorov *GG* nevedeli správne určiť: *H010.*
- Počet správnych odporúčaní len pomocou *GL*, ktoré sme kombináciou skupinových a globálnych vzorov *GG* nevedeli správne určiť: *H100.*
- Počet správnych odporúčaní získaných aj pomocou odporúčaní z *GL* aj *GR*: *H111+H110*.
- Teoreticky najvyšší možný počet správnych odporúčaní, ktoré by mohla metóda kombináciou skupinových a globálnych vzorov dosiahnuť. Ak by sa vedela vždy správne rozhodnúť či na odporúčanie položky pre konkrétne sedenie používateľa použiť skupinové vzory alebo globálne: *H111+H110+H101+H100+H011+ H010.*

8.3.2 Vyhodnotenie úspešnosti

V Tabuľke 7 uvádzame rozdiely medzi presnosťami jednotlivých metód. Vykonali sme tiež jednoduchý štatistický t-test medzi populáciami hodnôt výsledkov metriky *presnosť* všetkých vyhodnocovaných konfigurácií metód.

Tabuľka 7. Presnosti najlepších konfigurácií parametrov pre jednotlivé metódy (v %).

	P@1	P@2	P@3	P@4	P@5	P@10	P@15
E-LEARNING PORTÁL (ALEF)							
GG	50.71	51.26	51.73	51.96	52.41	52.62	52.13
GL	49.27	49.85	50.44	50.73	51.06	51.10	50.45
GR	32.41	32.93	33.21	33.24	33.57	34.09	33.40
GG - GL	1.44	1.41	1.29	1.23	1.35	1.52	1.68
GG -GR	18.30	18.33	18.52	18.72	18.84	18.53	18.73
SPRAVODAJSKÝ PORTÁL							
GG	64.88	53.68	55.68	55.66	54.91	51.23	42.54
GL	63.69	52.62	54.58	54.19	52.76	49.03	41.16
GR	13.06	10.86	15.43	19.88	23.07	11.89	5.98
GG - GL	1.19	1.06	1.10	1.47	2.15	2.20	1.38
GG - GR	51.82	42.82	40.25	35.78	31.84	39.34	37.28

Pre dataset ALEF metóda *GG* dosiahla signifikantne vyššiu presnosť oproti *GL* ($p < 0.0001$, $t=9.7153$, $df=7600$ a rozdiel stredných hodnôt je 1.65%) a tiež oproti *GR* ($p < 0.0001$, $t=103.2904$, $df=7600$ a rozdiel stredných hodnôt je 21.43%). Pre dataset SP metóda *GG* dosiahla signifikantne vyššiu presnosť oproti *GL* ($p < 0.0001$, $t=3.8976$, $df=12598$ a rozdiel stredných hodnôt je 0.86%) a tiež oproti *GR* ($p < 0.0001$, $t=239.6845$, $df=12598$ a rozdiel stredných hodnôt je 41.37%). Tieto výsledky súhlasia s hypotézou H_1, ktorú sme uviedli na začiatku kapitoly.

Ďalej sledujeme rôzne prieniky a zjednotenia vektorov úspešnosti odporúčania získaných pomocou metódy *GL*, *GR* a *GG* (Tabuľka 8) pre vybranú najlepšiu konfiguráciu z optimalizácie v predchádzajúcej časti 8.2.

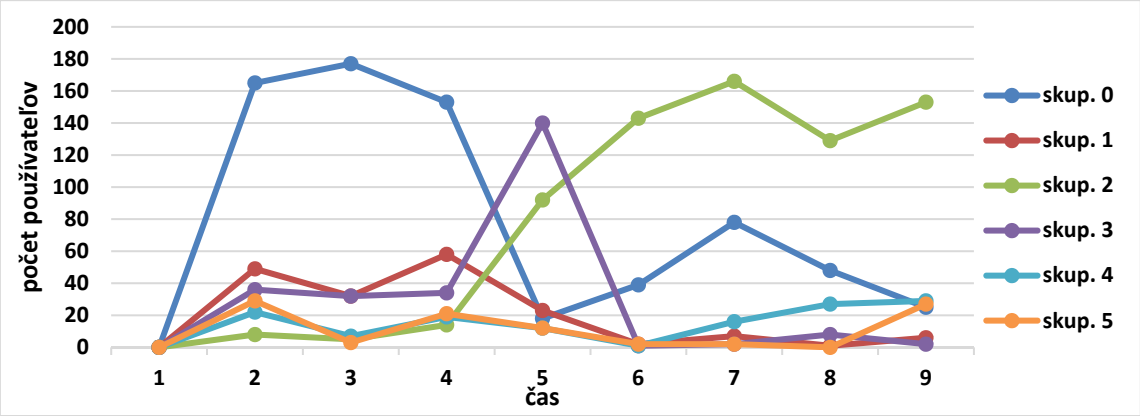
Tabuľka 8. Hodnoty *presnosti* (v %) pre rôzne počty odporúčaných položiek a rôzne prieniky metód GG, GR, GL.

Opis	Výpočet	ALEF							SPRAVODAJSKÝ PORTÁL						
		P@1	P@2	P@3	P@4	P@5	P@10	P@15	P@1	P@2	P@3	P@4	P@5	P@10	P@15
Hypotetická situácia, kedy je vybraný vždy lepší výsledok z GL alebo GR	H111+H110+H101+H100+H011+H010	55.14	57.02	58.40	59.06	59.88	63.91	60.72	66.79	55.92	59.33	60.65	61.00	54.65	43.26
GG	H101+H011+H001+H111	50.71	51.26	51.73	51.96	52.41	52.62	52.13	64.88	53.68	55.68	55.66	54.91	51.23	42.54
GL	H111+H110+H101+H100	49.27	49.85	50.44	50.73	51.06	51.10	50.45	63.69	52.62	54.58	54.19	52.76	49.03	41.16
GR	H111+H011+H110+H010	32.41	32.93	33.21	33.24	33.57	34.09	33.40	13.06	10.86	15.43	19.88	23.07	11.89	5.98
Len GR a nie GL	H011 + H010	5.86	7.17	7.97	8.33	8.82	9.96	10.27	3.10	3.30	4.74	6.46	8.24	4.47	2.10
Len GL a nie GR	H101+H100	22.72	24.09	25.19	25.82	26.31	26.98	27.32	53.72	45.06	43.90	40.78	37.93	41.61	37.28
Prienik GL a GR	H111+H110	26.55	25.77	25.24	24.91	24.74	24.12	23.13	9.97	7.55	10.68	13.41	14.83	7.42	3.88
Len GG a nie GR ani GL	H001	0.32	0.84	1.19	1.54	1.84	2.85	3.49	0.08	0.49	0.80	1.24	1.70	1.15	0.71
Len GR a nie GG ani GL	H010	2.63	3.41	3.95	4.16	4.40	5.02	5.08	1.29	1.36	1.92	2.59	3.05	1.26	0.58
Len GL a nie GG ani GR	H100	2.02	2.60	3.00	3.27	3.47	4.05	4.34	0.66	1.18	2.05	2.73	3.35	1.27	0.40
GG , GR aj GL	H111	26.45	24.92	23.78	23.14	22.60	20.63	18.54	9.94	7.37	10.20	12.49	13.45	6.53	3.44
GG a GL a nie GR	H101	20.70	21.28	21.69	22.03	22.17	21.60	20.82	53.06	43.88	41.85	38.05	34.58	40.34	36.88
GG a GR a nie GL	H011	3.23	3.72	3.92	4.08	4.29	4.66	4.71	1.80	1.94	2.83	3.87	5.19	3.21	1.52
GL a GR a nie GG	H110	0.10	0.59	0.89	1.19	1.40	2.09	2.41	0.03	0.18	0.48	0.92	1.38	0.89	0.45

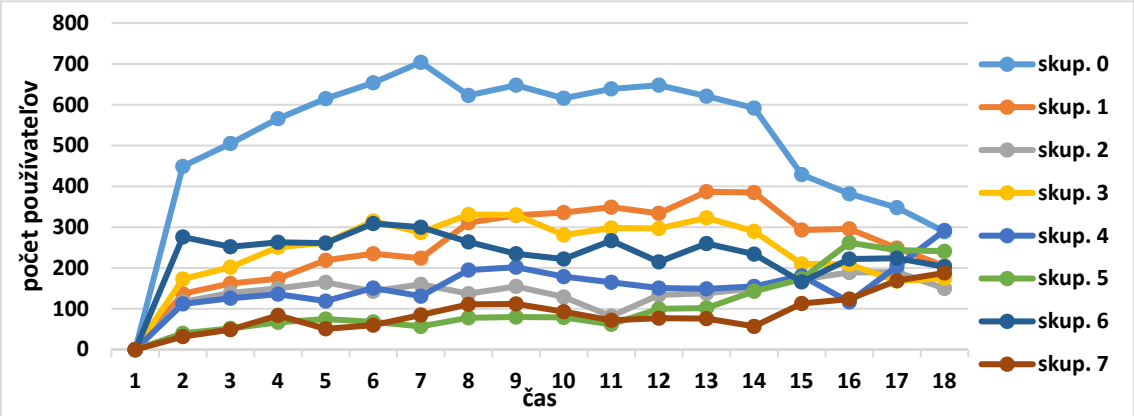
V Tabuľke 8 (1.riadok) prezentujeme hypotetickú situáciu kedy je použité ideálne kombinovanie globálnych a skupinových vzorov správania pre úlohu odporúčania, ktoré sa vie vždy správne rozhodnúť, ktoré množiny odporúčaní z *GL* a *GR* použiť. Nám sa podarilo dosiahnuť našou metódou presnosť odporúčaní lepšiu oproti *GR* a *GL*. V dôsledku navrhovaného spôsobu kombinácie vzorov boli tiež generované správne odporúčania aj v niektorých ďalších prípadoch, kedy negenerovali správne odporúčania ani skupinové ani globálne vzory samostatne (*H001*). Môžeme si tiež všimnúť výrazný prienik úspešných odporúčaní generovaných zároveň *GL* aj *GR* (*H110* + *H111*). Toto môže poukazovať na podobnú výpovednú hodnotu skupinových aj globálnych vzorov pri úlohe odporúčania. Počet správnych odporúčaní, ktoré dokázali vygenerovať len globálne vzory a skupinové vzory nie (*H100+H101*) je značne vyšší ako počet správnych odporúčaní, ktoré generovali len skupinové vzory (*H010+H011*). Toto je spôsobené tým, že v skupinách nevznikalo tak veľa silných vzorov (s vysokou podporou) ako v globálnom priestore, keďže aj príslušné dátové vzorky boli menšie a nie všetci používatelia boli zaradení do skupín. Sledovali sme, že presnosť odporúčania *GR* je vyššia v skupinách s väčším počtom aktívnych používateľov.

V pravidelných diskrétnych meraniach v čase spracovávania prúdu sme sledovali distribúciu počtu používateľov v skupinách, počty unikátnych vzorov v skupinách a presnosť jednotlivých metód. Merania boli vykonané pravidelne tesne pred vykonaním ďalšieho makrozhlukovania (9 meraní ALEF, 18 meraní v SP).

Počet používateľov v skupinách v ALEFe (Graf 8) kolíše výraznejšie ako v SP (Graf 9) čo svedčí o vyššej stabilite skupín v SP oproti ALEFu vďaka vhodnej abstrakcií akcií používateľov do kategórií spoločných pre všetkých používateľov a relevantných počas celého sledovaného obdobia. V ALEFe akcie reprezentujeme ako návštevy používateľa na konkrétnych vzdelávacích objektoch spojených s konkrétnym kurzom, ktorý má svojich účastníkov a obmedzené trvanie.

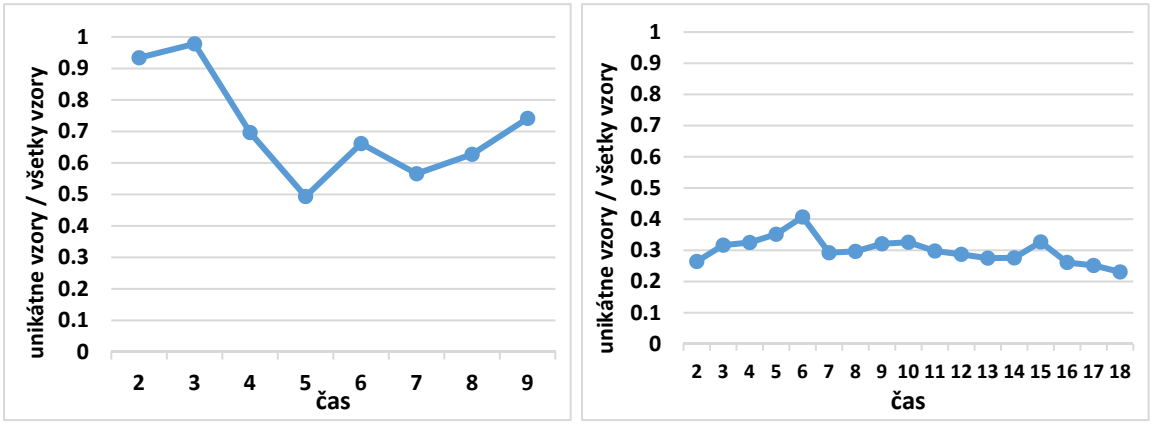


Graf 8. Počet používateľov v jednotlivých skupinách v datasete ALEF v pravidelných diskrétnych meraniach v čase počas spracovania dát.



Graf 9. Počet používateľov v jednotlivých skupinách v datasete SP v pravidelných diskrétnych meraniach v čase počas spracovania dát.

Na Grafe 10 sledujeme aký je pomer počtu unikátnych vzorov v skupinách ku počtu všetkých ostatných vzorov v skupine. Skupinový vzor považujeme za unikátny ak sa nevyskytuje v žiadnej inej skupine ani v množine globálnych vzorov. Tento pomer sa drží nad hranicou 50% v ALEFe a 20% v SP. Sledovali sme tiež, že tento pomer je rôzny pre rôzne skupiny. Napr. jedna skupina v SP obsahovala počas spracovania prúdu 60-90% unikátnych vzorov a iná len 7-20%. Absolútne počty unikátnych vzorov boli rôzne naprieč skupinami a rôzne v priebehu spracovania prúdu (od 1-100 v SP a 0-550 v ALEFe). Podrobne tieto výsledky uvádzame v Prílohe G. Tieto výsledky súhlasia so stanovenou hypotézou H_2, ktorá tvrdí, že skupinové vzory dokážu odhaliť nový pohľad na správanie používateľov v podobe unikátnych vzorov správania oproti globálnym vzorom správania.

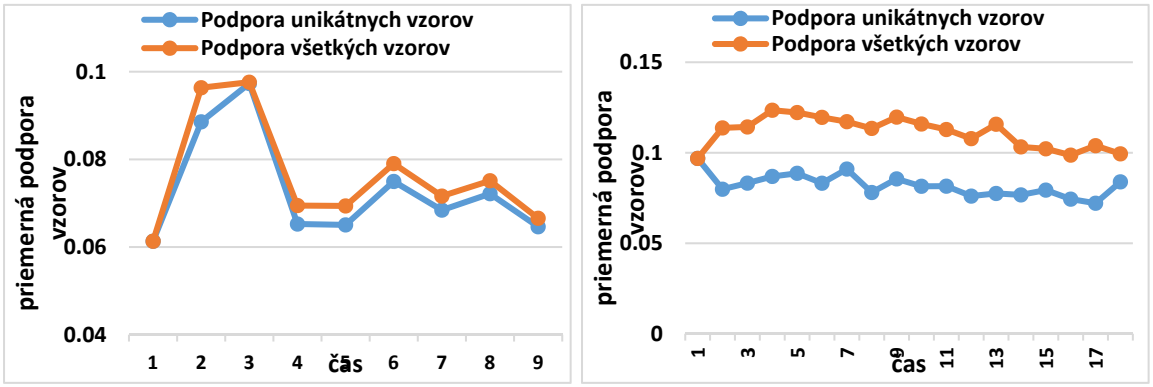


a) ALEF b) Spravodajský portál

Graf 10. Pomer počtu unikátnych vzorov v skupinách k celkovému počtu vzorov v pravidelných diskrétnych meraniach v čase počas spracovania dát. Dataset ALEF (a) a Spravodajský portál (b).

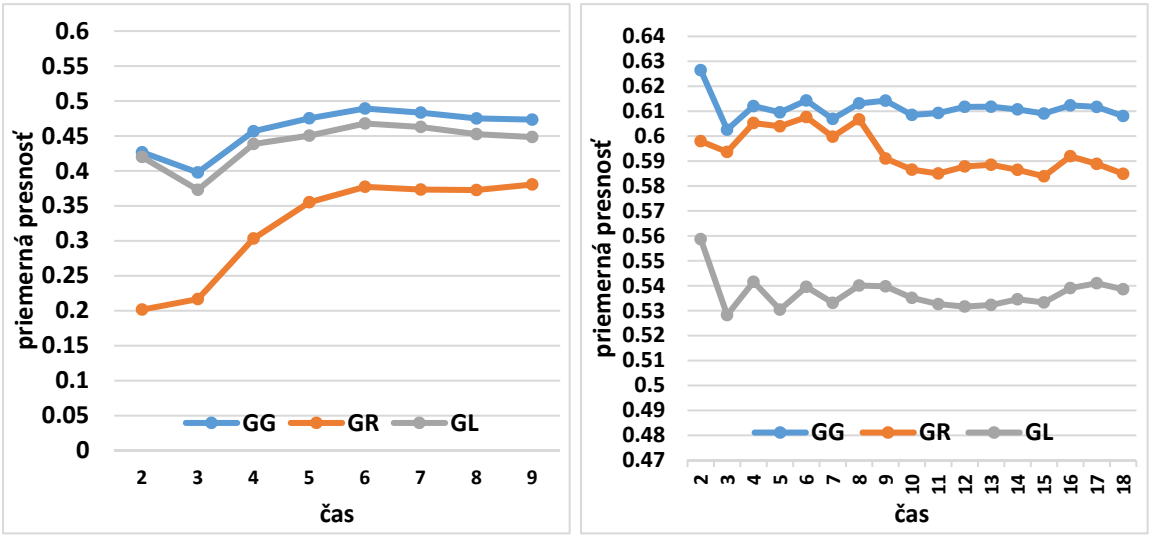
Na Grafe 11 vidíme, že hodnota priemernej podpory unikátnych vzorov je väčšinou o niečo málo menšia ako priemerná podpora všetkých vzorov. To znamená, že veľmi silné vzory (s vysokou podporou) sú často nachádzane duplicitne vo viacerých skupinách.

Okrem celkovej presnosti metód sme pozorovali aj ich úspešnosť vo vnútri jednotlivých identifikovaných skupín používateľov. Uvádzame priemernú presnosť vo vnútri skupín (Graf 12). Z týchto výsledkov možno vidieť že metóda *GG* dosahuje v oboch datasetoch najvyššiu presnosť. V datasete ALEF metóda *GL* dosahuje lepšiu presnosť ako *GR*, zatiaľ čo v datasete SP je situácia opačná. Toto je spôsobené najmä rozdielom v počte používateľov a dát o ich aktivite. V datasete SP je veľké množstvo používateľov, ktorí tak môžu byť rozdelení do kvalitnejších zhlukov. V datasete SP je malé množstvo používateľov, ktorí navyše vykonávajú špecifickejšie sedenia. Napriek tomuto obmedzeniu sú skupinové vzory v oboch prípadoch úspešne použité na zlepšenie výsledkov odporúčania.



a) ALEF b) Spravodajský portál

Graf 11. Priemerná podpora unikátnych vzorov a všetkých vzorov (aj s duplicitami) v diskrétnych meraniach v čase behu spracovania dát. Dataset e-learningový portál ALEF (a) a Spravodajský portál (b).



a) ALEF

b) Spravodajský portál

Graf 12. Priemerná presnosť odporúčania vo vnútri skupín používateľov v datasete ALEF (a) a Spravodajský portál (b). Presnosť je vypočítaná ako priemer presností vo vnútri jednotlivých skupín.

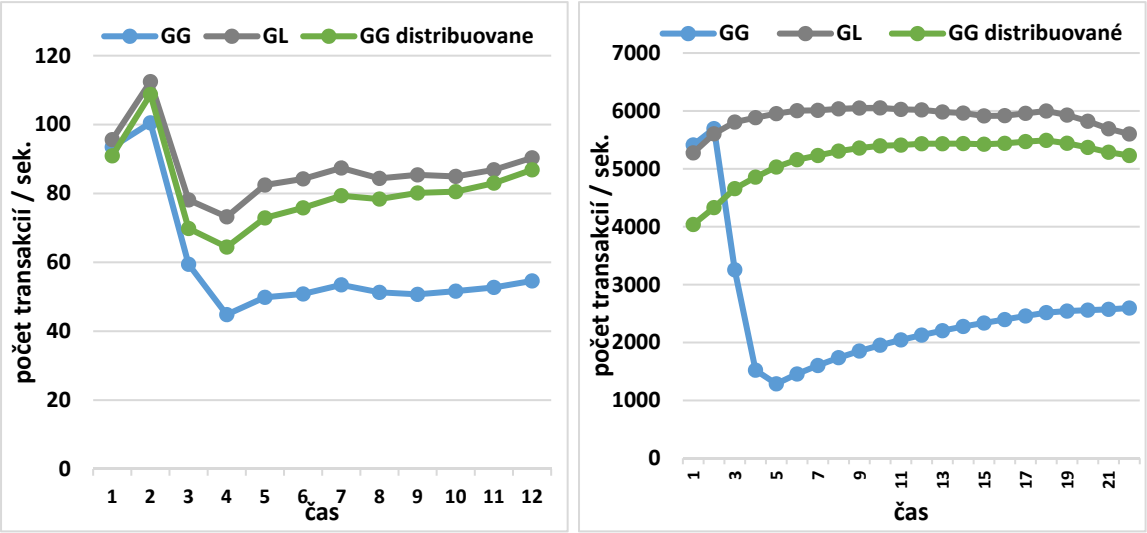
8.3.3 Vyhodnotenie rýchlosti spracovania prúdu dát

Okrem presnosti odporúčania hodnotíme tiež rýchlosť spracovania prúdu dát. Pod rýchlosťou spracovania myslíme počet používateľských sedení spracovaných metódou za časovú jednotku. Metóda *GG* zahŕňa zhlukovanie v prúde dát a hľadanie skupinových a globálnych vzorov správania zároveň. To prináša zvýšenú výpočtovú náročnosť oproti riešeniu, ktoré hľadá len globálne vzory správania (*GL*).

Aktuálnu rýchlosť (priemerný počet spracovaných transakcií za sekundu) logujeme v pravidelných časových intervaloch. Na Grafe 13 je možno vidieť porovnanie priemerných rýchlostí metód (bez ich aplikácie v úlohe odporúčania) *GL* a *GG* (so zhlukovaním a hľadaním skupinových vzorov) [10] a tiež metódy *GG* aplikovanej v distribuovanom prostredí pomocou technológie *Apache Storm* (pozri časť 7.2) z 10 nezávislých behov.

Z výsledkov je jasné, že zhlukovanie a hľadanie skupinových vzorov spôsobuje vyššiu výpočtovú a časovú náročnosť. Rozdiel medzi *GG* a *GL* hoci spočiatku má rastúci trend sa zdá byť na konci ustálený a teda v praxi udržateľný. Paralelizácia jednotlivých úloh tak ako sme ju navrhli a implementovali pomocou rámca Apache Storm (7.2) okrem toho významne znižuje tento rozdiel.

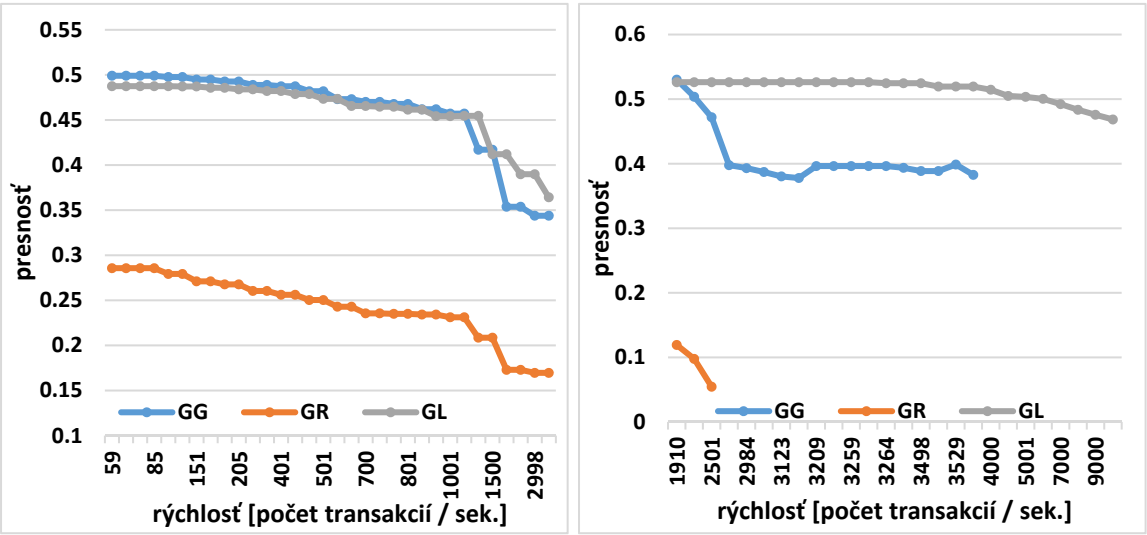
[10] Pokiaľ nie je uvedené inak, experimenty boli vykonané na výpočtovom zariadení s parametrami: Procesor: *Intel(R) Core(TM) i5-3210M CPU @ 2.50GHz, 2501 Mhz, 2 jadrá, 4 logické jadrá.* RAM: *8Gb.*



a) ALEF b) Spravodajský portál

Graf 13. Rýchlosť spracovania prúdu dát. Porovnanie metód *GL* a *GG* a paralelizované *GG* (implementované v distribuovanej topológií *Apache Storm*).

Okrem toho sme testovali účinok zvyšovania podmienky minimálnej požadovanej rýchlosti (vstupný parameter *mts*) na zmenu *presnosti* odporúčania. Na Grafe 14 je znázornené k akému znižovaniu presnosti odporúčania dochádza pri zvyšovaní nárokov na rýchlosť metódy *GG*, *GR*, a *GL*. V prípade datasetu ALEF sledujeme pomalé zmenšovanie a zánik rozdielu medzi *GG* a *GL*. V prípade datasetu SP vieme, že metóda *GL* je značne rýchlejšia a k zániku rozdielu medzi *GG* a *GL* dochádza prakticky ihneď po aplikovaní vyššej *mts* ako je skutočná rýchlosť spracovania.

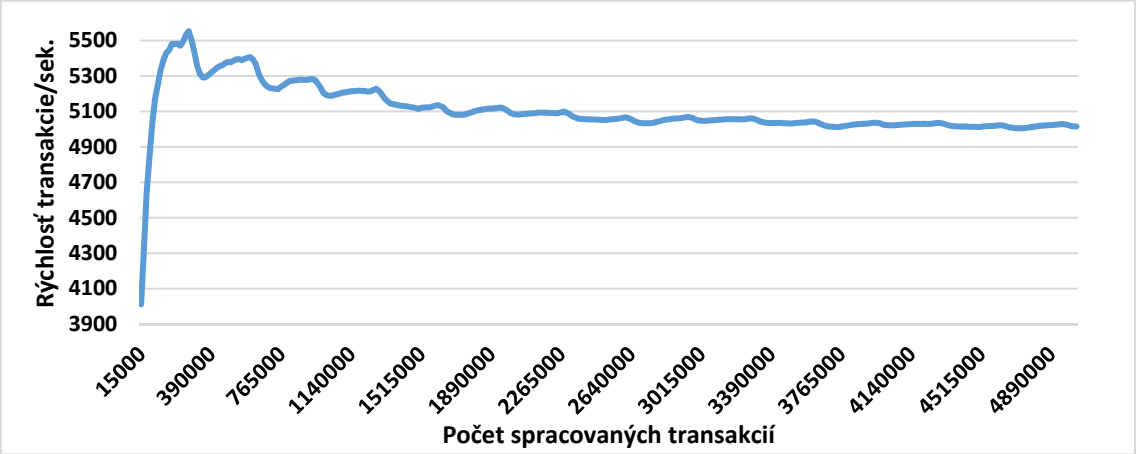


a) ALEF b) Spravodajský portál

Graf 14. Graf znázorňuje vzťah medzi zvyšujúcou sa rýchlosťou spracovania (na horizontálnej osi) a znižujúcou sa presnosťou odporúčania jednotlivých metód. Merania metód *GG* a *GR* sú zastavené pri maximálnej dosiahnutej rýchlosti.

V závislosti od požiadaviek aplikácie je teda možné obmedziť rýchlosť spracovania pomocou parametra *mts*. V takom prípade však môže byť prínos kombinácie skupinových vzorov s globálnymi v úlohe odporúčania záporný. Zvýšenie rýchlosti je preto vhodnejšie dosiahnuť použitím distribuovanej verzie *GG* (7.2). V prípade ešte vyššej požadovanej rýchlosti je vhodnejšie použiť inú metódu (napr. len *GL*).

Nakoniec sme vykonali jednoduché vytrvalostné testovanie. Metódu *GG* v distribuovanej verzií (7.2) bez aplikácie na úlohu odporúčania, sme aplikovali na umelo predĺžený dataset *SP* (spracovanie datasetu viacnásobne v slučke, bez vyčistenia dátových štruktúr po ukončení). Na Grafe 15 možno vidieť ako sa rýchlosť spracovania po čase ustálila, čo svedčí o schopnosti metódy vyrovnať sa s neustále pribúdajúcimi dátami.



Graf 15. Vytrvalostný test na umelo predĺženom datasete SP.

8.3.4 Porovnanie navrhnutej metódy voči existujúcim algoritmom

Rozhodli sme sa navrhovanú metódu porovnať s algoritmami, ktoré riešia podobné úlohy nad prúdom dát. Dôvody výberu algoritmu zhlukovania *CluStream* a hľadania vzorov *IncMine* pre našu metódu sme už sčasti vysvetlili v návrhu metódy (kapitola 6). Metódu sme navrhli tak aby bolo možné jednoducho integrovať rôzne algoritmy zhlukovania a hľadania vzorov správania, čím sa nám podarilo vytvoriť priestor pre ďalšie experimentovanie a porovnávanie s inými algoritmami pre jednotlivé časti metódy.

Porovnávame výsledky rôznych kombinácií vybraných algoritmov zhlukovania *CluStream* (Aggarwal C. C., Han, Wang, & Yu, 2003) a *ClusTree* (Kranen, Assent, Baldauf, & Seidl, 2011) s algoritmami dolovania frekventovaných vzorov *IncMine* (Quadrana, Bifet, & Gavaldà, 2015) , *EstDec+* (Shin, Lee, & Lee, 2014) a *CloStream* (Yen, Wu, Lee, Tseng, & Hsieh, 2011) (Tabuľka 9). Pre algoritmy *CluStream* a *IncMine* používame najlepšiu nájdenú konfiguráciu parametrov (8.2.6) a ich implementácie upravené z rámca *MOA* tak ako sme uviedli v kapitole 7. Ďalej používame implementáciu algoritmu *ClusTree* dostupnú v rámci *MOA* so základným nastavením parametrov, ktorú sme integrovali s rovnako nastaveným *k-means* makrozhlukovacím algoritmom ako pri algoritme *CluStream*. Algoritmy *CloStream* a *EstDec+* sme integrovali do nášho experimentálneho prostredia z rámca *SFPM* (Fournier-Viger, 2016). Algoritmus

CloStream nevyžaduje vstupné parametre. Algoritmus *EstDec+* má niekoľko vstupných parametrov, preto sme najskôr uskutočnili hľadanie najlepšieho nastavenia parametrov testovaním výsledkov pre rôzne kombinácie hodnôt (spôsobom opísaným v 8.2.1). Vybrali sme konfiguráciu, ktorá dosahovala najlepšiu presnosť pri zachovaní požiadavky na minimálnu rýchlosť (150 trans./s pre SP a 15 trans./s pre ALEF).

S algoritmom zhlukovania *CluStream* sme dosiahli mierne lepšie výsledky v presnosti než s algoritmom *ClusTree*. Algoritmus hľadania vzorov *IncMine,* pri zachovaní vysokej rýchlosti v porovnaní k ostatným algoritmom dosahuje najlepšie výsledky presnosti pre menšie počty odporúčaných položiek (≤ 5).

Z výsledkov v Tabuľke 9 je zrejmé, že nami vybraná kombinácia algoritmov v kombinácií s nižšou dimenzionalitou akcií používateľa (ako je to v prípade datasetu SP) je vhodná na použitie v prúde dát a najmä na odporúčanie menšieho počtu položiek.

Tabuľka 9. Porovnanie presnosti a rýchlosti pre rôzne kombinácie použitých algoritmov zhlukovania a hľadania vzorov správania voči nami vybraným algoritmom *Clustream* a *IncMine*.

Zhlukovaci algoritmus	Algoritmus hľadania vzorov	Rýchlosť	Presnosť odporúčania [%]						
			P@1	P@2	P@3	P@4	P@5	P@10	P@15
Spravodajský portál									
CluStream	**IncMine**	2077.5	64.88	53.68	55.68	55.66	54.91	51.23	42.54
Clustree	IncMine	2563.8	64.86	53.68	55.62	55.63	54.85	51.13	42.35
Bez zhlukovania	IncMine	3526.6	63.68	52.62	54.58	54.20	52.79	49.05	41.15
CluStream	EstDec	424.6	49.40	42.79	47.39	48.66	48.30	46.75	42.48
Clustree	EstDec	520.0	49.09	42.60	47.08	48.36	48.17	45.35	40.64
Bez zhlukovania	EstDec	547.9	48.75	42.44	46.13	46.76	45.72	43.67	39.74
CluStream	CloStream	415.4	49.33	36.66	43.12	46.04	47.80	60.45	65.84
Clustree	CloStream	436.3	48.86	36.63	42.84	45.65	47.37	60.03	65.79
Bez zhlukovania	CloStream	706.7	46.90	34.42	41.80	44.97	46.32	59.51	65.53
ALEF									
CluStream	**IncMine**	49.4	50.71	51.26	51.73	51.96	52.41	52.62	52.13
Clustree	IncMine	46.9	50.40	50.40	50.39	50.58	50.71	49.33	47.72
Bez zhlukovania	IncMine	76.5	49.14	49.21	49.19	49.43	49.46	48.07	46.57
CluStream	EstDec	8.6	27.66	30.98	33.48	35.08	36.52	39.98	40.58
Clustree	EstDec	8.6	27.65	31.98	34.27	35.87	37.11	40.31	40.63
Bez zhlukovania	EstDec	15.2	25.00	28.48	30.28	32.00	33.01	35.86	36.21
CluStream	CloStream	8.1	39.45	40.89	41.09	41.69	42.11	43.69	44.54
Clustree	CloStream	8.3	43.31	44.99	45.89	46.74	47.10	47.24	46.61
Bez zhlukovania	CloStream	14.7	39.00	41.61	41.81	42.97	43.54	43.69	44.13

Navrhovanú metódu pracujúcu s prúdom dát sme dodatočne porovnali s nami implementovanou metódou, ktorá využíva klasický algoritmus na hľadanie vzorov *FP-Growth* (Han, Pei, & Yin, 2000) (Tabuľka 10). Nájdené vzory správania používa na odporúčanie zaujímavých stránok používateľom rovnakým spôsobom ako nami navrhovaná metóda. *FP-Growth* je algoritmus, ktorý funguje dobre a pomerne rýchlo na statických datasetoch (viac v Prílohe A). Našim zámerom bolo ukázať, že navrhovaná metóda dokáže dosiahnuť porovnateľnú presnosť ako tradičné offline prístupy s oveľa

vyššou rýchlosťou. Datasety sme rozdelili na trénovaciu a testovaciu časť v pomere 1:1 a porovnali presnosť odporúčania obidvoch metód na druhej polovici datasetu. Sledovali sme že metóda využívajúca algoritmus *FP-Growth* potrebuje vybudovať model, ktorý sa v dynamicky meniacom prostredí rýchlo stane neaktuálnym. Z prvej polovice datasetu vygenerovala zbytočne veľké množstvo vzorov (9887 v SP a 76758 v ALEFe) čo následne spomaľovalo aj proces ich využitia na odporúčanie. Keďže naša metóda si udržiava vždy menší počet vzorov, ktoré sú však veľmi aktuálne a tiež špecifickejšie pre konkrétnych používateľov, dokáže dosiahnuť lepšiu presnosť a odporúčať efektívnejšie v online čase. Z výsledkov je zrejmé, že nami navrhovaná metóda je vhodná na použitie v dynamickom prúde dát, dokáže reagovať na zmeny v správaní používateľov a môže byť využitá na efektívne odporúčanie.

Tabuľka 10. Porovnanie navrhovanej metódy voči statickému algoritmu FP-Growth. *ms* označuje zvolenú hranicu minimálnej podpory a *ews* zvolenú veľkosť vyhodnocovacieho okna (počet položiek sedenia použitých na hľadanie vzorov).

Metóda	Počet sedení v trénovacej množine	Veľkosť testovacej množiny	ms	ews	Presnosť [%]					Celkový čas spracovania [s] (hľadanie vzorov + odporúčanie)
					P@1	P@2	P@3	P@4	P@5	
SPRAVODAJSKÝ PORTÁL										
FP-Growth	167127	167130	0.0004	1	45.81	42.66	31.31	24.31	19.86	14319.99
Navrhovaná metóda (GG)	evaluácia od 167130-teho sedenia		0.05	1	63.27	52.50	54.99	55.78	55.97	87.11
E-LEARNING PORTÁL (ALEF)										
FP-Growth	12296	12298	0.02	2	27.55	24.74	23.63	21.90	20.60	18190.90
Navrhovaná metóda (GG)	evaluácia od 12298-eho sedenia		0.05	2	51.40	51.81	51.76	51.57	51.52	240.73

9 Záver a zhodnotenie

Problematika hľadania a aplikácie vzorov správania používateľov webového sídla je široká a má svoje špecifické problémy. Znalosti o správaní používateľov, ktoré môžu byť získané vo forme vzorov správania môžu znamenať konkurenčnú výhodu, základ pre správne biznis rozhodnutia a tiež zvyšovanie úspešnosti metód predikcie správania používateľov či odporúčania.

Vzory správania nie sú pevne definovaným pojmom a vo webovom sídle si ich môžeme predstavovať ako rôzne skutočnosti (sekvencie často po sebe navštívených stránok, typické stránky ktorými začínajú sedenia používateľov, postupnosti navštevovaných sekcií webového sídla, priemerné dĺžky návštev stránok, priemerné dĺžky sedení, a mnoho ďalších identifikovaných pravidelne sa opakujúcich atribútov súvisiacich so správaním používateľa vo webovom sídle). My sme sa zamerali na vzory správania reprezentované ako frekventované množiny a sekvencie akcií používateľov.

Opísali sme viaceré existujúce metódy hľadania vzorov správania a ich aplikácie v úlohách predikcie či odporúčania. Ich spoločným problémom je, že obsahujú výpočtovo náročnú časť zabezpečujúcu dolovanie vzorov. Takto nájdené vzory sú síce presné, ale možnosť ich použitia v dynamickom prostredí, kde sa správanie používateľov mení s pribúdajúcimi novými dátami, je obmedzená. Nájdené vzory správania sú navyše príliš široké a všeobecné, preto dobre nevystihujú správanie užších a silno špecifických skupín používateľov.

V práci sme navrhli metódu, ktorá využíva algoritmy schopné pracovať s prúdom dát, identifikuje vzory správania používateľov v prúde dát reprezentujúcich akcie používateľov. Špecifikom metódy je, že hľadá vzory správania v dvoch úrovniach. Identifikuje tzv. *globálne vzory* správania spoločné pre všetkých používateľov webového sídla a tzv. *skupinové vzory* typické pre dynamicky identifikované užšie skupiny používateľov webového sídla. Navrhovaná metóda dokáže rýchlo a flexibilne odhaľovať aktuálne záujmy používateľov webového sídla a zmeny ich záujmov v čase. Súčasťou návrhu metódy je aj návrh spôsobu kombinácie globálnych a skupinových vzorov správania pre účel ich aplikácie v úlohe odporúčania zaujímavých položiek používateľom.

Vzory správania reprezentujeme ako uzavreté frekventované množiny. Frekventované množiny sú jednoduchšou reprezentáciou vzorov správania oproti napr. frekventovaným sekvenciám alebo asociačným pravidlám. Vďaka tomu sú ale algoritmy ich hľadania efektívnejšie. Uzavreté frekventované množiny sú navyše kompletnou a neredundantnou reprezentáciou všetkých frekventovaných množín, čo značne napomáha k zmenšeniu prehľadávaného priestoru.

Používatelia sú zhlukovaní na základe modelov používateľa do skupín, pre ktoré sú identifikované *skupinové vzory správania.* Modely používateľa sú aktualizované priebežne, vďaka čomu sú skupinové vzory správania nachádzané v aktuálnych skupinách a metóda je schopná reagovať na zmeny v rozdeleniach používateľov do skupín.

Navrhnutú metódu sme implementovali v dvoch prototypoch. Vhodne zvolenou štruktúrou sme vytvorili priestor pre porovnávanie rôznych algoritmov zhlukovania a hľadania vzorov. Implementovali sme tiež prototyp pomocou rámca *Apache Storm* v

ktorom sú úlohy hľadania vzorov správania pre jednotlivé skupiny používateľov vykonávané paralelne, čo výrazne znižuje réžiu spojenú s hľadaním týchto špecifických vzorov.

Navrhnutú metódu sme overili na dvoch datasetoch s odlišnými charakteristikami z domény e-learningu a spravodajstva.

Keďže navrhovaná metóda ma pomerne veľké množstvo vstupných parametrov, rozhodli sme sa najskôr uskutočniť ich optimalizáciu. Hľadali sme takú konfiguráciu hodnôt parametrov pri ktorej bola maximalizovaná presnosť odporúčania pri zachovaní požiadaviek na rýchlosť metódy.

Metódu sme overili na úlohe odporúčania a sledovali sme metriku *presnosť.* Navrhovaná metóda dosiahla štatisticky signifikantne lepšie výsledky ako referenčné metódy, ktoré používali na odporúčanie iba globálne vzory, resp. iba skupinové vzory správania. Ďalej sme navrhnutú metódu porovnali s jej verziami používajúcimi rôzne algoritmy zhlukovania a hľadania vzorov správania, kde sa ukázal náš výber algoritmov a ich použitie ako najúspešnejšie.

Hodnotili sme tiež rýchlosť spracovania dát metódou. Zaujímalo nás najmä akú réžiu prináša hľadanie skupinových vzorov správania. Zistili sme, že táto réžia je konštantná a teda v praxi udržateľná. Pomocou navrhnutej paralelizácie je tiež výrazne redukovaná.

Počas tohto projektu sme pracovali s rôznymi nápadmi. Pre metódu opisovanú v tejto práci sme sa rozhodli práve z dôvodu, že reaguje na výzvy rýchleho spracovania dát ako prúdu a ponúka schopnosť flexibilnej reakcie na zmeny v správaní používateľov. V budúcnosti je možné existujúce riešenie ďalej rozširovať o nové možnosti. Prototyp sme implementovali tak aby bolo možné integrovať a skúšať aj iné algoritmy zhlukovania a hľadania vzorov správania. Reprezentácia vzorov správania môže byť obohatená o ďalšie atribúty ako napr. o časový údaj pri položkách v rámci sedenia používateľa, ktorý môže poukazovať na relevantnosť položky. Možno tiež experimentovať s ďalšími reprezentáciami akcií používateľa, okrem návštev stránok/kategórií webového sídla to môžu byť napr. vyplnenie dotazníka, odoslanie komentára, pridanie/odobranie položky z košíka v e-obchode a pod. Jeden používateľ by mohol byť zaradený do viacerých skupín. Nielen podľa spoločného správania ale i iných atribútov ako napr. použitého zariadenia, alebo podľa demografie (veku, vzdelania, pohlavia, príjmu, lokality). Uzly dolujúce vzory správania lokálnej skupiny v rámci distribuovanej verzie metódy by mohli byť efektívne umiestnené blízko lokality konkrétnej skupiny. Pre veľké webové sídla by takýchto skupín mohlo byť naozaj veľa, nielen tie ktoré sa dynamicky identifikovali, ale rôznorodé segmenty používateľov, ktorých správanie chceme analyzovať.

10 Literatúra

Agarwal, R. C., Aggarwal, C. C., & Prasad, C. C. (2001). A Tree Projection Algorithm for Generation of Frequent Item Sets. *Journal of Parallel and Distributed Computing, 61*(3), 350-371.

Agarwal, R. C., Aggarwal, C. C., & Prasad, V. (2000). Depth first generation of long patterns. *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining* (s. 108--118). Boston, MA, USA: ACM.

Aggarwal, C. C., Bhuiyan, M. A., & Hasan, M. A. (2014). Frequent Pattern Mining Algorithms:A Survey. In C. C. Aggarwal, & J. Han, *Frequent Pattern Mining* (s. 19-61). Springer International Publishing. doi:10.1007/978-3-319-07821-2

Aggarwal, C. C., Han, J., Wang, J., & Yu, P. S. (2003). A framework for clustering evolving data streams. *Proceedings of the 29th international conference on Very large data bases. 29*, s. 81-92. Berlin, Germany: VLDB Endowment.

Aggarwal, C. C., Han, J., Wang, J., & Yu, P. S. (2004). A Framework for Projected Clustering of High Dimensional Data Streams. *Proceedings of the Thirtieth International Conference on Very Large Data Bases. 30*, s. 852-863. Toronto, Canada: VLDB Endowment.

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. *Proc. 20th int. conf. very large data bases, VLDB. 1215*, s. 487-499. Santiago de Chile, Chile: VLDB Endowment.

Agrawal, R., & Srikant, R. (1995). Mining sequential patterns. *Proceedings of the Eleventh International Conference on Data Engineering* (s. 3–14). Washington, DC, USA: IEEE.

Anandhi, D., & Irfan Ahmed, M. S. (2014). An improved web log mining and online navigational pattern prediction. *Research Journal of Applied Sciences, Engineering and Technology, 8*(12), 1472-1479.

Bayardo, J., & Roberto, J. (1998). Efficiently mining long patterns from databases. *ACM Sigmod Record, 27*(2), 85--93.

Bieliková, M., Šimko, M., Barla, M., Tvarožek, J., Labaj, M., Móro, R., . . . Ševcech, J. (2014). ALEF: From Application to Platform for Adaptive Collaborative Learning. In *Recommender Systems for Technology Enhanced Learning: Research Trends and Applications* (s. 195-225). New York, NY: Springer New York.

Bifet, A., Holmes, G., Kirkby, R., & Pfahringer, B. (May 2010). MOA: Massive Online Analysis. *The Journal of Machine Learning Research, 11*, 1601-1604.

Burdick, D., Calimlim, M., & Gehrke, J. (2001). MAFIA: A maximal frequent itemset algorithm for transactional databases. *Data Engineering, 2001. Proceedings. 17th International Conference on* (s. 443-452). IEEE.

Calders, T., Dexters, N., Gillis, J. J., & Goethals, B. (January 2014). Mining frequent itemsets in a stream. *Information Systems, 39*, 233-255.

Cao, F., Ester, M., Qian, W., & Zhou, A. (2006). Density-Based Clustering over an Evolving Data Stream with Noise. *Proceedings of the 2006 SIAM International Conference on Data Mining* (s. 328-339). SIAM.

Chang, J. H., & Lee, W. S. (2003). Finding recent frequent itemsets adaptively over online data streams. *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining* (s. 487-492). Washington, DC, USA: ACM.

Chen, Y., & Tu, L. (2007). Density-based Clustering for Real-time Stream Data. *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (s. 133-142). New York, NY, USA: ACM.

Cheng, J., Ke, Y., & Ng, W. (2008). Maintaining frequent closed itemsets over a sliding window. *Journal of Intelligent Information Systems, 31*(3), 191-215.

Chi, Y. a. (2006). Catch the moment: Maintaining closed frequent itemsets over a data stream sliding window. *Knowledge and Information Systems, 10*(3), 265-294.

Chi, Y., Wang, H., Yu, P. S., & Muntz, R. R. (2004). Moment: Maintaining closed frequent itemsets over a stream sliding window. *ICDM'04. Fourth IEEE International Conference on Data Mining* (s. 59-66). Brighton, UK: IEEE.

Etminani, K., Delui, A. R., Yenehsari, N. R., & Rouhani, M. (2009). Web Usage Mining: Discovery of the user's navigational patterns using SOM. *Networked Digital Technologies, 2009. NDT'09. First International Conference on* (s. 224-249). Ostrava, Czech Republic: IEEE.

Facca, F. M., & Lanzi, P. L. (2003). Recent Developments in Web Usage Mining Research. In *Data Warehousing and Knowledge Discovery: 5th International Conference* (s. 140-150). Berlin: Springer Berlin Heidelberg. doi:10.1007/978-3-540-45228-7_15

Fournier-Viger, P. (07. 11 2016). Dostupné na Internete: SPMF: An Open-Source Data Mining Library: http://www.philippe-fournier-viger.com/spmf/index.php

Fu, Y., Shih, M.-Y., Creado, M., & Ju, C. (2002). Reorganizing web sites based on user access patterns. *International Journal of Intelligent Systems in Accounting, Finance & Management, 11*(1), 39-53.

Han, J., Kamber, M., & Pei, J. (2012). 6 - Mining Frequent Patterns, Associations, and Correlations: Basic Concepts and Methods. In J. Han, M. Kamber, & J. Pei, *Data mining: concepts and techniques* (s. 243 - 278). Boston: Elsevier.

Han, J., Pei, J., & Yin, Y. (2000). Mining Frequent Patterns Without Candidate Generation. *ACM Sigmod Record* (s. 1-12). ACM.

Han, J., Pei, J., Mortazavi-Asl, B., Chen, Q., Dayal, U., & Hsu, M.-C. (2000). FreeSpan: frequent pattern-projected sequential pattern mining. *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining* (s. 355-359). Boston, MA, USA: ACM.

Herder, E. (2007). *An Analysis of User Behavior on the Web-Understanding the Web and its Users.* VDM Verlag.

Huntington, P., Nicholas, D., & Jamali, H. R. (2008). Website usage metrics: A re-assessment of session data. *Information Processing and Management, 44*(1), 358-372.

Jalali, M., Mustapha, N., Nasir Sulaiman, M., & Mamat, A. (2010). WebPUM: A Web-based recommendation system to predict user future movements. *Expert Systems With Applications, 37*, 6201-6212.

Kosala, R. a. (2000). Web Mining Research: A Survey. *ACM SIGKDD Explorations Newsletter, 2*(1), 1-15.

Kranen, P., Assent, I., Baldauf, C., & Seidl, T. (2011). The ClusTree: indexing micro-clusters for anytime stream mining. *Knowledge and Information Systems, 29*(2), 249-272.

Krempl, G., Spiliopoulou, M., Stefanowski, J., Žliobaite, I., Brzeziński, D., Hüllermeier, E., . . . Sievi, S. (2014). Open Challenges for Data Stream Mining Research. *ACM SIGKDD Explorations Newsletter, 16*(1), 1-10.

Kum, H.-C., Pei, J., Wang, W., & Duncan, D. (2003). Approxmap: Approximate mining of consensus sequential patterns. *SDM Conference* (s. 311-315). San Francisco, CA: SIAM.

Lee, V. E., Jin, R., & Agrawal, G. (2014). Frequent Pattern Mining in Data Streams. In C. C. Aggarwal, & J. Han, *Frequent Pattern Mining* (s. 199-220). Springer.

Li, H., Zhang, N., & Chen, Z. (2012). A Simple but Effective Maximal Frequent Itemset Mining Algorithm over Streams. *JSW, 7*(1), 25-32.

Li, H.-F., Ho, C.-C., Kuo, F.-F., & Lee, S.-Y. (2006). A new algorithm for maintaining closed frequent itemsets in data streams by incremental updates. *Data Mining Workshops, 2006. ICDM Workshops 2006. Sixth IEEE International Conference on* (s. 672--676). IEEE.

Liraki, Z., & Harounabadi, A. (2015). Predicting the Users ' Navigation Patterns in Web , using Weighted Association Rules and Users ' Navigation Information. *International Journal of Computer Applications, 110*(12), 16-21.

Makker, S., & Rathy, R. (2011). Web Server Performance Optimization using Prediction Prefetching Engine. *International Journal of Computer Applications (0975--8887), 23*(9).

Mao, G., Wu, X., Zhu, X., Chen, G., & Liu, C. (2007). Mining maximal frequent itemsets from data streams. *Journal of Information Science, 33*(3), 251-262.

Mobasher, B., Dai, H., Luo, T., Sun, Y., & Zhu, J. (2000). Integrating Web Usage and Content Mining for More Efective Personalization. *International Conference on Electronic Commerce and Web Technologies* (s. 165-176). London, UK: Springer.

Pei, J., Han, J., & Mao, R. (2000). CLOSET: An Efficient Algorithm for Mining Frequent Closed Itemsets. *ACM SIGMOD workshop on research issues in data mining and knowledge discovery*, *4*, s. 21-30.

Pei, J., Han, J., Mortazavi-asl, B., Pinto, H., Chen, Q., Dayal, U., & chun Hsu, M. (2001). Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth. *proceedings of the 17th international conference on data engineering* (s. 215-224). Washington, DC, USA: IEEE Computer Society.

Perkowitz, M., & Etzioni, O. (1997). Adaptive web sites: An AI challenge. *IJCAI International Joint Conference on Artificial Intelligence*, 16-21.

Pinto, H., Han, J., Pei, J., Wang, K., Chen, Q., & Dayal, U. (2001). Multi-dimensional sequential pattern mining. *Proceedings of the tenth international conference on Information and knowledge management* (s. 81-88). Atlanta: ACM.

Quadrana, M., & Mestre, R. G. (2012). Methods for Frequent Pattern Mining in Data Stream within the MOA System. Universitat Politecnica de Catalunya.

Quadrana, M., Bifet, A., & Gavaldà, R. (2015). An Efficient Closed Frequent Itemset Miner for the MOA Stream Mining System. *Ai Communications, 28*(1), 143-158.

Savasere, A. a. (1995). *An efficient algorithm for mining association rules in large databases.* Georgia Institute of Technology.

Shen, W., Wang, J., & Han, J. (2014). Sequential Pattern Mining. In C. C. Aggarwal, & J. Han, *Frequent Pattern Mining* (s. 261-281). Springer.

Shin, S. J., Lee, D. S., & Lee, W. S. (2014). CP-tree: An adaptive synopsis structure for compressing frequent itemsets over online data streams. *Information Sciences, 278*, 559-576.

Silva, J., Faria, E., Barros, R., Hruschka, E., de Carvalho, A., & Gama, J. (2013). Data stream clustering: A survey. *ACM Computing Surveys (CSUR), 46*(1), 13.

Sisodia, D. S., & Verma, S. (2012). Web usage pattern analysis through web logs: A review. *Computer Science and Software Engineering (JCSSE), 2012 International Joint Conference on* (s. 49-53). Bangkok, Thailand: IEEE.

Song, G., Yang, D., Cui, B. a., Liu, Y., & Xie, K. (2007). CLAIM: An efficient method for relaxed frequent closed itemsets mining over stream data. *International Conference on Database Systems for Advanced Applications* (s. 664--675). Bangkok, Thailand: Springer.

Spiliopoulou, M. a. (2003). A Framework for the Evaluation of Session Reconstruction Heuristics in Web-Usage Analysis. *INFORMS J. on Computing, 15*(2), 171-190.

Srikant, R., & Agrawal, E. (1996). Mining Sequential Patterns: Generalization and Performance Improvements. *5th International Conference on Extending Database Technology (EDBT '96)* (s. 3-17). Avignon, France: Springer.

Srivastava, M., Garg, R., & Mishra, P. K. (2014). Preprocessing Techniques in Web Usage Mining: A Survey. *International Journal of Computer Applications (0975 – 8887), 97*(18), 1-9.

Teng, W. G., Chang, C. Y., & Chen, M. S. (2005). Integrating web caching and web prefetching in client-side proxies. *IEEE Transactions on Parallel and Distributed Systems, 16*(5), 444-455.

Tsymbal, A. (2004). The Problem of Concept Drift: Definitions and Related Work. *Computer Science Department, Trinity College Dublin, 106*(2).

Tzvetkov, P., Yan, X., & Han, J. (2005). TSP: Mining top-k closed sequential patterns. *Knowledge and Information Systems, 7*(4), 438-457.

Vellingiri, J., Kaliraj, S., Satheeshkumar, S., & Parthiban, T. (2015). A Novel Approach for User Navigation Pattern Discovery and Analysis for Web Usage Mining. *Journal of Computer Science, 11*(2), 372-382.

Wang, J., & Han, J. (2004). BIDE: Efficient mining of frequent closed sequences. *Data Engineering, 2004. Proceedings. 20th International Conference on Data Engineering* (s. 79-90). Boston, Massachusetts: IEEE.

Yan, X., Han, J., & Afshar, R. (2003). CloSpan: Mining: Closed sequential patterns in large datasets. *Proceedings of the 2003 SIAM International Conference on Data Mining* (s. 166-177). San Francisco, CA, USA: SIAM.

Yen, S.-J., Lee, Y.-S., Wu, C.-W., & Lin, C.-L. (2009). An efficient algorithm for maintaining frequent closed itemsets over data stream. *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems* (s. 767-776). Springer.

Yen, S.-J., Wu, C.-W., Lee, Y.-S., Tseng, V. S., & Hsieh, C.-H. (2011). A fast algorithm for mining frequent closed itemsets over stream sliding window. *Fuzzy Systems (FUZZ), 2011 IEEE International Conference on. 86*, s. 996-1002. Taipei, Taiwan: IEEE.

Yu, X., & Korkmaz, T. (2015). Heavy path based super-sequence frequent pattern mining on web log dataset. *Artif. Intell. Research, 4*(2), 1–12.

Zaki, M. J. (2000). Scalable Algorithms for Association Mining. *IEEE Trans. on Knowl. and Data Eng., 12*(3), 372-390.

Zaki, M. J. (2001). SPADE: An Efficient Algorithm for Mining Frequent Sequences. *Machine Learning, 42*(1), 31-60.

Zaki, M. J., & Gouda, K. (2003). Fast vertical mining using diffsets. *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining* (s. 326-335). Washington, DC, U.S.A: ACM.

Zaki, M. J., & Hsiao, C.-J. (2002). CHARM: An efficient algorithm for closed itemset mining. *Proceedings of the 2002 SIAM International Conference on Data Mining* (s. 457-473). Arlington: SIAM.