Slovenská technická univerzita v Bratislave

Fakulta informatiky a informačných technológií

FIIT-XXXXX-XXXXX

Bc. Tomáš Chovaňák

ROZPOZNÁVANIE VZOROV SPRÁVANIA POUŽÍVATEĽOV WEBOVÉHO SÍDLA

Diplomový projekt 3

Študijný program: Informačné systémy  
Študijný odbor: 9.2.5.Informačné systémy   
Miesto vypracovania: Ústav informatiky, informačných systémov a softvérového inžinierstva, FIIT STU  
Vedúci práce: Ing. Ondrej Kaššák   
  
december 2016

**Zadanie:**

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.

Návrh zadania diplomovej práce

Návrh zadania 2. strana

Slovenská technická univerzita v Bratislave  
**FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ**

**Rozpoznávanie vzorov správania používateľov webového sídla**

Študijný program: Informačné systémy

Autor: Bc. Tomáš Chovaňák

Vedúci diplomovej práce: Ing. Ondrej Kaššák

december 2016

Vzory správania môžeme chápať ako typické a opakujúce sa črty správania používateľov pri návšteve webového sídla. V tejto práci reprezentujeme vzory správania ako frekventované množiny akcií vykonávaných v sedeniach používateľov. Častým zdrojom znalostí o správaní používateľa sú webové logy a v nich zachytené akcie, ktoré vykonali používatelia počas návštevy webového sídla, spájané do používateľských sedení. Celý proces spracovania webových logov, hľadania vzorov a ich analýzy býva označovaný aj ako dolovanie znalostí z dát o používaní Webu (ang. *Web Usage Mining*). Nájdené vzory správania môžu slúžiť napríklad na vytváranie odporúčaní, predikciu zámerov používateľa s využitím na ukladanie stránok do vyrovnávajúcej pamäte, podporu zmeny dizajnu webového sídla, či celkové pochopenie správania používateľov. Väčšina existujúcich riešení tohto procesu hľadá vzory správania zo statických webových logov v množine všetkých používateľov webového sídla.

Táto práca reaguje na súčasný trend personalizácie Webu a zameriavania sa na potreby jednotlivcov a tiež na výzvu dolovania znalostí z rýchlych prúdov dát.Navrhujeme proces, ktorý dokáže spracovávať sedenia používateľov ako prúd dát a objaviť vzory správania, ktoré nehovoria len o správaní globálnej komunity používateľov, ale aj o aktuálnom správaní a zmenách správania menších komunít. Vyhodnocujeme prínos kombinovania skupinových a globálnych vzorov v ich aplikácií na úlohu odporúčania a tiež sledujeme ako dokáže metóda odhaľovať unikátne správanie špecifických skupín používateľov v doméne elektornického výučbového systému a novinového portálu.

Slovak University of Technology Bratislava  
**FACULTY OF INFORMATICS AND INFORMATION TECHNOLOGIES**

**Recognition of Web user’s behavioral patterns**

Degree Course: Information systems

Author: Bc. Tomáš Chovaňák

Supervisor: Ing. Ondrej Kaššák

2016, December

Behavioral patterns can be understood as typical and repeating features of user’s behavior during their visit of website. In this work we represent behavioral patterns as frequent itemsets of actions frequently taken by user’s in user sessions. Frequent source of knowledge about user’s behavior are web logs and actions taken by user’s during their visits to website stored there, joined to sessions. Whole process of processing web logs, finding behavioral patterns and their analysis is known as Web Usage Mining. Found behavioral patterns can be used for example to create recommendations, predicting user’s intentions (which can be used to cache predicted pages), as support for website design change or complex understanding of website user’s behavior. Most of existing methods of Web Usage Mining search for behavioral patterns common for whole set of web site users in static web logs.

This work responds to actual trend of Web personalization and focusing on needs of individual users. It also responds to challenge to mine knowledge from fast streaming data. We propose solution that is able to process data about user sessions as streaming data and search for behavioral patterns telling us not only about behavior of global community of users, but also about actual behavior and changes in behavior of smaller communities of users. We evaluate contribution of combining global and group behavioral patterns in their application to recommendation task. We also observe way this method is able to detect unique behavior of specific users groups in domain of e-learning system and news portal.

Obsah

[1 Úvod 1](#_Toc472758751)

[2 Dolovanie z dát na webe a vo webovom sídle 3](#_Toc472758752)

[2.1 Získavanie a predspracovanie dát 5](#_Toc472758753)

[2.2 Využitia a aplikácie dát o používaní Webu 8](#_Toc472758754)

[2.3 Sumarizácia a diskusia 10](#_Toc472758755)

[3 Existujúce prístupy hľadania a reprezentácie vzorov správania v statických datasetoch 11](#_Toc472758756)

[3.1 Dolovanie frekventovaných množín 12](#_Toc472758757)

[3.2 Dolovanie frekventovaných sekvencií 15](#_Toc472758758)

[3.3 Iné reprezentácie vzorov správania a prístupy k ich získavaniu 18](#_Toc472758759)

[3.4 Sumarizácia a diskusia 20](#_Toc472758760)

[4 Existujúce prístupy hľadania a reprezentácia vzorov správania v prúde dát 21](#_Toc472758761)

[4.1 Dolovanie frekventovaných množín v prúde dát 23](#_Toc472758762)

[4.2 Zhlukovanie nad prúdom dát 27](#_Toc472758763)

[4.3 Existujúce rámce na spracovávanie kontinuálneho prúdu dát 29](#_Toc472758764)

[4.4 Sumarizácia a diskusia 30](#_Toc472758765)

[5 Navrhovaná metóda reprezentácie, extrakcie a aplikácie vzorov správania nad prúdom dát 31](#_Toc472758766)

[5.1 Základné pojmy 31](#_Toc472758767)

[5.2 Proces spracovania transakcií 35](#_Toc472758768)

[5.3 Sumarizácia vstupných parametrov metódy 42](#_Toc472758769)

[6 Implementácia metódy 44](#_Toc472758770)

[6.1 Architektúra riešenia 44](#_Toc472758771)

[7 Overenie metódy 47](#_Toc472758772)

[7.1 Spôsob overenia jednotlivých aspektov a opis použitých dát 47](#_Toc472758773)

[7.2 Vyhodnotenie úspešnosti odporúčania pre jednotlivé skupiny parametrov v datasete ALEF 56](#_Toc472758774)

[7.3 Výsledné vyhodnotenie pre dataset ALEF 64](#_Toc472758775)

[8 Záver a zhodnotenie 69](#_Toc472758776)

[Zoznam použitej literatúry 70](#_Toc472758777)

# Úvod

Snaha získať čo najviac a čo najvýstižnejších znalostí  o správaní používateľov webových sídiel je dnes nepochybne veľmi aktuálnou témou. Tak ako to dobrí obchodníci iste vedia, pochopenie zákazníka je nesmierne dôležité. Web je 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 väčšinou 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 na dobre 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 produkt. Alebo spravodajský portál, kde producent chce získať čo najväčšiu čítanosť a konzument chce získať informácie.

Správanie človeka nie je deterministické, ale jeho chovanie veľakrát podlieha naučeným vzorom, ktoré si často ani neuvedomuje. Existujú skupiny ľudí, ktorí sa vo viacerých situáciách správajú podobne. Asi však neexistujú dvaja ľudia, ktorí sa správajú celkom rovnako. Príkladom podobného správania skupiny ľudí môže byť spôsob čítania novín. Niektorí ľudia napr. čítajú noviny „odzadu”. Nie je to však tak vždy. Pre niekoho je takýto vzor správania silnejší a pre niekoho zas slabší a pre niekoho neplatí vôbec. Vzory správania môžeme chápať napríklad ako často opakujúce sa množiny akcií používateľov. Častým zdrojom znalostí o správaní používateľa sú webové logy a v nich zachytené akcie, ktoré vykonali používatelia počas návštevy webového sídla, spájané do používateľských sedení. V tejto práci vychádzame práve z takto získavanej implicitnej spätnej väzby používateľa. Sedenie používateľa môžeme reprezentovať ako množinu akcií a vzory správania ako frekventované množiny, či sekvencie akcií nájdené v takto upravených logoch.

Cieľom tejto práce je v súlade so súčasným trendom personalizácie Webu a zameriavania sa na potreby jednotlivcov objaviť čo najkvalitnejšie vzory správania, ktoré nebudú hovoriť len o správaní globálnej komunity používateľov, ale aj správaní menších komunít a jednotlivcov k nim patriacich. Dosiahnuť sa to snažíme kombinovaním vzorov správania získaných pre skupiny používateľov s podobným správaním s globálnymi vzormi správania spoločnými pre celú komunitu používateľov webového sídla.

V prvej časti tejto práce sa venujeme najskôr podrobnejšej analýze problematiky dolovania dát na Webe a vo webovom sídle v kapitole 2, kde prehľadne vysvetľujeme samotný pojem dolovania dát na Webe ako súčasť širšieho výskumu a potom opisujeme aj jednotlivé kroky procesu dolovania dát na Webe a tiež možné aplikácie nájdených znalostí.

Následne v kapitole 3 opisujeme pojem vzor správania a jeho rôzne reprezentácie v statických datasetoch,  priblížujeme niektoré konkrétne spôsoby dolovania vzorov správania v statických dátach reprezentovaných ako frekventované množiny a frekventované sekvencie. V kapitole 4 sa takisto venujeme spôsobom dolovania vzorov správania, ale z prúdu dát. Opisujeme špecifické požiadavky na algoritmy, ktoré spracovávajú prúd dát. Na záver tejto kapitoly ešte približujeme problematiku zhlukovania v prúde dát a existujúce rámce, ktoré vedia spracovať prúd dát a ponúkajú sadu algoritmov dolovania znalostí z prúdu dát. Tieto znalosti neskôr využívame v kapitole 5, ktorý obsahuje návrh metódy, ktorej hlavným cieľom je hľadať nielen vzory správania spoločne pre celú komunitu používateľov, ale zároveň 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 rýchlo generované veľké množstva dát tým, že dáta spracováva ako prúd a spĺňa požiadavky, ktoré sú na takýto typ spracovania dát kladené. V kapitole 6 opisujeme implementáciu nášho riešenia. V kapitole 7 opisujeme metodológiu a realizáciu experimentov v ktorých overujeme prínos metódy v aplikácií na odporúčanie a vyhodnocujeme tiež vzťahy medzi globálnymi a skupinovými vzormi. Na záver v kapitole 8 zhŕňame podstatné poznatky, ktoré sme v tejto práci získali a opisujeme možný ďalší smer práce.

# Dolovanie znalostí z dát na webe a vo webovom sídle

Dolovanie znalostí z dát na Webe (ang. Web Mining) znamená 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 informácií, extrakcia informácií, strojové učenie. Pri interakcií s Webom môžu byť techniky dolovania dát na webe použité na riešenia rôznych problémov ako sa uvádza aj v (Kosala, 2000):

* hľadanie relevantných informácií (špecifickej informácie),
* získavanie nových znalostí z dostupných informácií ,
* personalizácia informácií,
* odhaľovanie záujmov používateľa a zákazníka.

Okrem toho tiež v práci (Kosala, 2000) rozdeľuje problém dolovania dát na Webe do štyroch úloh :

* 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.

Existujú tri kategórie dolovania Webu podľa časti Webových zdrojov v ktorej sa dolovanie znalostí uskutočňuje (Sisodia & Verma, 2012) :

* Dolovanie znalostí z obsahu Webu (ang. Web content mining). Predstavuje dolovanie znalostí zo samotného obsahu Webových dokumentov (teda vnútornej štruktúry dokumentov).
* Dolovanie znalostí zo štruktúry Webu (ang. Web structure mining). Predstavuje hľadanie modelu štruktúry prepojení medzi stránkami a zvlášť hľadanie zaujímavých vzťahov v tomto modeli.
* Dolovanie znalostí z dát o používaní Webu (ang. Web usage mining, skr. WUM). Predstavuje dolovanie znalostí v dátach generovaných správaním používateľov Webu. Teda dát, ktoré zaznamenávajú interakciu používateľa s Webom. Hľadanie vzorov správania používateľov je teda úlohou vyplývajúcou práve z tejto kategórie dolovania Webu a preto sa jej venujeme podrobnejšie aj v ďalších častiach tejto práce.

WUM je zaujímavou aplikáciou techník dolovania v dátach na získavanie znalostí o správaní používateľov vo webovom sídle. Proces aplikácie WUM na personalizáciu webového sídla 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 to, že vstupom nie sú subjektívne hodnotenia používateľov (explicitná spätná väzba), ale je založený na implicitnej spätnej väzbe, ktorá predstavuje objektívne skutočnosti ako sú akcie používateľa v systéme (Mobasher, Dai, Luo, Sun, & Zhu, 2000).

WUM môže byť považovaný za proces pozostávajúci z 5-tich fáz (Sisodia & Verma, 2012) :

1. Zbieranie dát z Webových zdrojov,
2. Príprava a predspracovanie dát. Bližšie rozoberáme v časti 2.1.
3. Hľadanie vzorov v správaní používateľov. Viacerým zaujímavým metódam hľadania a reprezentovania vzorov sa venujeme v ďalších kapitolách 3 a 4.
4. Analýza a validácia nájdených vzorov a použitie v rôznych aplikáciách. Bližšie rozoberáme v časti 2.2.

## Získavanie a predspracovanie dát

Prvým krokom procesu WUM ako sme si ho uviedli v predchádzajúcej kapitole je získavanie dát o správaní návštevníkov webového sídla. 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 webovým sídlom (Sisodia & Verma, 2012).

Dátové sady, ktoré používame na overenie navrhovaných metód v tejto práci už prešli viacerými druhmi predspracovania, z toho dôvodu sa otázke predspracovania dát nevenujeme s takým dôrazom na detaily ako v ďalších kapitolách. Skôr sa snažíme poskytnúť všeobecnejší pohľad na rôzne možnosti predspracovania dát.

V tejto práci pracujeme najmä s dátami, ktoré vznikli spracovaním logov webových serverov. Konzorcium W3C poskytuje štandardný odporúčaný formát webových logov[[1]](#footnote-2), ale samozrejme existujú aj iné proprietárne formáty. Použitie štandardného formátu umožňuje využiť niektoré existujúce analytické nástroje, ktoré dokážu tento formát spracovávať[[2]](#footnote-3). Nazbierané dáta zaznamenané vo webových logoch typicky zdieľajú najmä tieto informácie: dátum, čas, IP klienta, autorizácia klienta, IP servera, port servera, metóda požiadavky, HTTP status kód, referenčná adresa, host, používateľský agent, čas vykonania akcie, počet prenesených byteov 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 potom vieme jednoduchšie nachádzať špecifickejšie vzory (Sisodia & Verma, 2012):

* Prístupové logy – zachytávajú detailne 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.

Najmä kvôli obmedzeniam daným špecifikáciou protokolu HTTP sú údaje v bežných webových logov nekompletné a neumožňujú jednoznačnú identifikáciu používateľa a teda ani jeho používateľských sedení. Vo svojej práci Spiliopoulou a kol. (Spiliopoulou, 2003) klasifikuje 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 bez identifikovania používateľov a ich sedení. Reaktívny prístup sa teda snaží transformovať existujúce webové logy, identifikovať používateľov a ich sedenia. Kvôli tomu, že tento prístup nevyžaduje žiaden špecializovaný softvér, ale využíva len všeobecné serverové 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 WUM teda identifikáciu používateľov a ich sedení využíva explicitné mechanizmy. Na identifikáciu používateľov je možné použiť cookies. Problém s cookies je, že používatelia môžu mať v prehliadači vypnuté ich používanie a v tom prípade je nemožné ich identifikovať. Webový server môže mať tiež implementovaný vlastný mechanizmus identifikácie používateľských sedení (Huntington, Nicholas, & Jamali, 2008). Problémom proaktívneho prístupu je, že používateľ sa musí prihlásiť a nainštalovať softvér, ktorý zabezpečí logovanie. Okrem toho sú tieto metódy niekedy dokonca zakázané ako spyware.

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 uvedené 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, suplementárne požiadavky napr. na obrazový obsah stránok alebo rôzne skripty (Srivastava, Garg, & Mishra, 2014).

#### Identifikácia používateľov

Pri väčšine webových serverov sú používatelia anonymní a problémom je, že nevieme presne určiť priradenie logovaných akcií k týmto používateľom.

Pri reaktívnom prístupe je jednou zo stratégií identifikácie používateľov je aproximácia identity používateľa z IP adries, operačného systému zariadenia z ktorého sa pripája a prehliadača, ktorý používa. Táto stratégia však samozrejme nedokáže presnejšie identifikovať konkrétneho používateľa 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 používateľa. V tomto prípade však treba získať trénovacie dáta , ktoré majú explicitne uvedené id používateľa. Ďalšou možnosťou identifikácie používateľov je sprístupnenie obsahu webového sídla až po autentifikácií používateľa, čo je však mnoho krát odradzujúcim prvkom. Ďalej je možné identifikovať používateľov pomocou cookies, tu ale existujú scenáre v ktorých takýto prístup nefunguje ako napr. prehliadač, ktorý má zakázané cookies, vymazanie cookies používateľom. (Srivastava, Garg, & Mishra, 2014)

#### Identifikácia používateľských sedení

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

#### Kompletizácia ciest

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

## Využitia a aplikácie dát o používaní Webu

Posledným krokom procesu WUM je použitie a aplikácia získaných znalostí o správaní používateľov, ktoré ďalej označujeme aj ako vzory správania a v ďalších kapitolách približujeme tento pojem z hľadiska ako vzor správania reprezentujeme v tejto práci. Všeobecným cieľom WUM je zbierať zaujímavé informácie o správaní používateľov a použiť ich na vylepšenie webového portálu z ich hľadiska (Facca & Lanzi, 2003). V tejto podkapitole opisujeme jednotlivé oblasti aplikácie vzorov správania získaných v procese WUM podľa (Facca & Lanzi, 2003).

**Personalizácia Webu**

Pomocou porovnávania aktuálneho správania s nájdenými vzormi správania je možné využiť vzory správania na predikciu ďalších krokov používateľa, odporúčanie produktov a položiek, či inú adaptáciu webového sídla podľa aktuálnych potrieb používateľa. Existuje viacero prác zaoberajúcich sa aplikáciou vzorov správania na personalizáciu, ktoré sa líšia najmä spôsobom hľadania a reprezentácie vzorov správania. V ohľade na túto aplikáciu je potrebné sa vo WUM procese zaoberať najmä identifikáciou používateľov a ich sedení v rámci fázy predspracovania. V rámci použitia na odporúčanie je zas kľúčové sa sústrediť na spôsob výberu vzorov z ktorých budú generované odporúčania podľa aktuálneho správania používateľa.

Príkladom je metóda WebPUM (Jalali, Mustapha, Nasir Sulaiman, & Mamat, 2010) založená na reprezentácií vzorov správania ako komponentov získaných po aplikácií algoritmu delenia grafu na graf reprezentujúci prepojenia medzi stránkami webového sídla, ktorých váha je odvodená od pohybu používateľov vo webovom sídle v ich sedeniach. Získané vzory správania používajú na generovanie odporúčaní podľa aktuálneho správania používateľov.

Ďalšia metóda na predikciu krokov používateľov je opísaná v práci (Liraki & Harounabadi, 2015). Je to sofistikovaný systém využívajúci Fuzzy C-means zhlukovanie na hľadanie vzorov správania reprezentovaných ako asociačné pravidlá. Vyhodnocujú úspešnosť použitia vzorov správania na predikciu ďalších krokov používateľa.

Ďalšou zaujímavou prácou je (Anandhi & Irfan Ahmed, 2014), kde predstavujú 3 rôzne príspevky k zlepšeniu konkrétnych krokov WUM procesu. Prvým je návrh vhodnejšej heuristiky v rámci fázy predspracovania na identifikáciu sedení používateľov. Druhým príspevkom je použitie zhlukovacieho algoritm DBSCAN na hľadanie vzorov správania. Ten je založený na hustote dátových bodov, čo zvyšuje pravdepodobnosť nájdenia aj menej frekventovaných vzorov, ktoré by napr. pri použití algoritmu hľadania asociačných pravidiel s definovanou hranicou minimálnej podpory boli ignorované. Tretím prínosom je návrh použitia invertovaného indexu pre efektívnejšiu online predikciu správania používateľa.

**Ukladanie stránok do dočasnej pamäti a prednačítavanie**

To, že sú vzory správania vhodným zdrojom pre metódy predikcie správania používateľa je možné využiť okrem odporúčania položiek aj na zvýšenie výkonnosti a času odozvy webového servera tým, že sa predikované položky na základe aktuálneho správania používateľa, ktoré s vysokou pravdepodobnosťou v ďalších krokoch navštívi 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 sofistikovaný algoritmus, ktorý tieto pravidlá efektívne využíva na ukladanie do dočasnej pamäte integrovaný s prednačítavaním stránok na strane klienta. V rámci tohto algoritmu tiež navrhli sofistikovanú funkciu ohodnotenia profitu z prípadného prednačítania konkrétneho objektu uvažujúce mnohé aspekty (napr. veľkosť objektu, počet referencií na objekt, spoľahlivosť použitého asociačného pravidla), vďaka ktorej vedia určiť či daný objekt 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 pre účely zlepšovania použiteľnosti webového sídla. Získané znalosti o správaní používateľa môžu byť tak 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 informáciam 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, ktoré opisujú je, že často navštevované stránky klasifikované ako obsahové umiestňujú automaticky vyššie v hierarchickej štruktúre. Malými zmenami štruktúry dosiahnú postupnú adaptáciu štruktúry webového sídla potrebám 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, ako napríklad rozhodnutia o segmentácií trhu. Existujú nástroje podporujúce takéto rozhodovanie od malých lacných nástrojov analyzujúcich webové logy (ako napr. WebLog Expert [[3]](#footnote-4)) po zložitejšie riešenia integrované aj v rámci CRM systémov (napr. WebTrends [[4]](#footnote-5)).

## Sumarizácia

V tejto časti sme opísali, čo znamená dolovanie znalostí z dát na webe, z akých krokov pozostáva a aké má praktické využitie. Z podkategórií dolovania dát na webe nás ďalej v tejto práci bude zaujímať najmä dolovanie znalostí z dát o používaní Webu (ang. Web usage mining a skr. WUM). WUM je proces, ktorý je možné automatizovať a pozostáva z troch hlavných krokov: predspracovanie dát, hľadanie vzorov správania a ich následná aplikácia na rôzne účely. V aplikáciach , ktoré sme opísali sú často oddelené kroky predspracovania a hľadania vzorov, ktoré sú vykonávané v rámci offline komponentu, a následná aplikácia nájdených vzorov vykonávaná v rámcií online komponentu. Tu sa vynára 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 ako potenciálne nekonečný prúd dát. Práve na túto výzvu sme sa tiež rozhodli v tejto práci reagovať.

Výhodou vzorov správania 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 sidle (Anandhi & Irfan Ahmed, 2014). Sú teda vynikajúcim zdrojom znalostí pre analýzu správania používateľov.

Samotná aplikácia nájdených vzorov môže byť rôzna, uviedli sme niekoľko prác, kde opisujú jednak rôzne metódy hľadania vzorov správania a tiež rôzne spôsoby ich využitia.

# Existujúce prístupy hľadania a reprezentácie vzorov správania v statických datasetoch

Zatiaľ čo predspracovanie dát je viac-či menej veľmi podobné pri rôznych metódach a aplikáciách WUM, rôzne techniky dolovania dát môžu byť aplikované s cieľom nájsť vzory správania používateľov vo webovom sídle (Sisodia & Verma, 2012).

Vzory správania nie sú nejakým pevne definovaným pojmom a vo webovom sídle si ich môžeme predstavovať ako rôzne skutočnosti napr. sekvencie často po sebe navštívených stránok, typické stránky ktorou 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.

Odteraz v tejto práci vzory správania používateľov webového sídla rozumieme ako v literatúre bežné uvádzané tzv. frekventované vzory (ang. *frequent pattern*) získané z dát zachytávajúcich správanie používateľov webového sídla (najčastejšie ako klasické webové logy). Ako sa uvádza v práci (Han, Kamber, & Pei, 2012) frekventovaný vzor zachytáva často opakujúce sa vzťahy v dátach. Tieto frekventované vzory môžu mať rôznu podobu – všeobecne to môžu byť často spolu sa vyskytujúce sa neusporiadané podmnožiny z množiny všetkých položiek, usporiadané subsekvencie celých sekvencií akcií, či iné subštruktúry celkovej štruktúry. Frekventované vzory sú dôležitým vstupom pre ďalšie úlohy dolovania v dátach ako hľadanie asociačných pravidiel (ktoré môžu byť tiež chápané ako reprezentácia vzorov správania), klasifikácia, zhlukovanie a ďalšie, čo robí z úlohy ich hľadania veľmi dôležitú oblasť výskumu.

Dôležitým rozdelením frekventovaných vzorov je podľa toho či zachovávajú sekvenčnosť vykonaných akcií alebo nie. Podľa toho možno frekventované vzory, rozdeliť na (Han, Kamber, & Pei, 2012):

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

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

## Dolovanie frekventovaných množín

Nech je databáza transakcií, Nech je množina všetkých položiek v . Položkou môže byť napríklad záznam o navštívení stránky webového sídla používateľom zachytený vo webových logoch, alebo čokoľvek iné napr. položka v košíku používateľa v internetovom obchode. Transakcia môže predstavovať napríklad sedenie používateľa, alebo košík používateľa v internetovom obchode. Transakcia je teda množinou položiek. Platí . Frekventovaná množina je akákoľvek množina s hodnotou podpory vyššou ako vopred určená hranica minimálnej podpory (ang. *minimum support threshold*). Nech početnosť 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 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 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.* Tiežplatí, 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 *B* má rovnakú hodnotu podpory ako *A*. Nech *C* je množina všetkých frekventovaných uzavretých množín. Nech *F* je množina všetkých frekventovaných vzorov. Aj pri *C* platí podobne ako pri maximálnych frekventovaných množinách, že veľkosť *C* je stále výrazne menšia ako *F.* Presnejšie platí :  *.*Tiež platí, že vieme zostrojiť množinu *F* z *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), ktorý má však vážne problémy s výpočtovou náročnosťou. K nemu existuje viacero algoritmov, ktoré sa ho snažia optimalizovať. Algoritmus Apriori a jeho varianty opisujeme podrobnejšie v prílohe A.
* **Algoritmy založené na stromovej štruktúre.** Problém generovania frekventovaných množín je ekvivalentný problému generovania stromovej štruktúry tzv. lexikografického stromu. Koreň stromu je prázdny a uzly stromu reprezentujú lexikograficky usporiadané množiny. Nech je množina korešpondujúca s uzlom stromu. Potom rodičom uzla je množina . 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 lexikograficky strom. Často sa líšia spôsobom konštrukcie tohto stromu. Algoritmus *Apriori* využíva stratégiu do šírky a lexikografický strom možno uvažovať len implicitne oproti napr. algoritmu *TreeProjection* (Agarwal, Aggarwal, & Prasad, 2001), ktorý existuje vo verzií so stratégiou prehľadávania do šírky aj do hĺbky a explicitne využíva lexikografický strom zobrazujúci proces generovania frekventovaných množín. Niektoré ďalšie algoritmy takisto existujú v modifikáciach podľa toho akú stratégiu prehľadávania stromu používajú (napr. Eclat a modifikácia dEclat (Zaki & Gouda, 2003). Jednou z výhod prístupu hľadania do hĺbky je v prípade dolovania dlhých maximálnych frekventovaných množín ich skoré nachádzanie. Explicitné použitie lexikografického stromu je nápomocné pri vizualizovaní stratégií prehľadávania kandidátnych frekventovaných množín a pochopeniu prínosu  rôznych algoritmov.
* **Algoritmy využívajúce vertikálny formát dát**. Nech *tid* označuje identifikátor transakcie, *tidlist* označuje množinu identifikátorov transakcie a *itemset* označuje množinu položiek. Základnou myšlienkou je zefektívnenie počítania podpory pomocou transformácie z horizontálneho formátu dát: *<tid,* *itemset>* na vertikálny formát dát *<itemset, tidlist>,* kde *tidlist* je zoznam transakcií v ktorých sa nachádza daná množina. 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). Podrobnejší opis algoritmu *Eclat*, ktorý patrí do tejto skupiny uvádzame v prílohe A.
* **Algoritmy založené rekurzívnom raste prípon vzorov.** Na rozdiel od väčšiny ostatných prístupov kde sa hľadajú vzory v lexikografickom strome budovaného postupne z predpôn jednotlivých lexikograficky usporiadaných množín, je tento prústup založený na postupnom rozširovaní prípon lexikograficky usporiadaných frekventovaných množín. Prehľadávanie v prístupe s rastom prípon vzorov nie je principíálne odlišné od klasického prehľadávania s rastom predpôn vzorov (napríklad *TreeProjection* algoritmus) s rozdielom, že položky nie sú usporiadané od tej s najmenšou podporou po najväčšiu ale naopak. 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. 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 redundantné. Ako sme už uviedli maximálne a uzavreté frekventované množinysú kompaktnou reprezentáciou väčšieho množstva frekventovaných množín. Algoritmy z tejto skupiny sa zameriavaju 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). V prílohe A bližšie opisujeme algoritmus *CHARM*.

## Dolovanie frekventovaných sekvencií

Úloha dolovania frekventovaných sekvencií bola prvýkrát definovaná v práci (Agrawal & Srikant, 1995). Dôležitým predpokladom je, že spracovávané dáta majú záznamy, ktoré sú sekvenčného charakteru. Príkladom môžu byť transakcie vykonané zákazníkmi elektronického obchodu či sekvencie DNA. Uvádzame definíciu problému dolovania frekventovaných sekvencií tak ako je uvedená v (Agrawal & Srikant, 1995).

Nech *C* je kolekcia všetkých položiek v dátach. Položkou môže byť napríklad záznam o navštívení stránky webového sídla používateľom zachytený vo webových logoch. Nech je neprázdnou množinou položiek, ktoré sa vyskytujú v dátach spoločne (pri sebe v sekvencií). Ak daná množina obsahuje napríklad položky a  tak ju môžeme zapísať aj takto: . Teda množinu položiek môžeme zapísať ako . Sekvencia je zoradeným zoznamom takýchto množín položiek. Teda kde označuje v poradí i-tu množinu položiek. je subsekvenciou ak existujú také indexy, že . Nech je databáza sekvencií. Každá sekvencia by mala obsahovať minimálne množinu a tiež identifikátor používateľa, ktorý danú transakciu vykonal a čas uskutočnenia transakcie.

Úlohou je nájsť všetky frekventované sekvencie nachádzajúce sa v  databáze *D.* Výpočet podpory frekventovanej sekvencie (vzoru) je vlastne výpočtom frekvencie výskytov danej sekvencie v *D.*  Sekvencia je považovaná za frekventovanú (a teda možno ju označiť ako 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 . Ak neexistuje žiadna nadsekvencia sekvenčného vzoru *α* 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).

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

Tabuľka 1 Príklad databázy sekvencií z (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 tak *s* má hodnotu podpory 2 a je teda sekvenčným vzorom s dĺžkou 3. Sekvencia je zas príkladom uzavretého sekvenčného vzoru keďže neexistuje taká nadsekvencia z *D*, ktorá by mala rovnakú hodnotu podpory (Shen, Wang, & Han, 2014).

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**. Ten hovorí, že všetky podmnožiny frekventovaných sekvencií sú tiež frekventované sekvencie. Ďalej možno algoritmy založené na tomto princípe rozdeliť podľa toho aký formát dát využívajú:
  + **algoritmy využívajúce horizontálny formát dát** (pozri Tabuľka 1). Patria tu napr. algoritmy *AprioriAll,* ktorý bol navrhnutý v práci (Agrawal & Srikant, 1995) ako jedno z prvých riešení problému hľadania frekventovaných sekvencií a 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). V prílohe B uvádzame podrobnejší opis k algoritmu GSP.
  + **algoritmy využívajúce vertikálny formát dát**:

.

Patria tu napr. algoritmy *SPADE* (Zaki M. J., 2001)*, SPAM* (Ayres, Flannick, Gehrke, & Yiu, 2002). V prílohe B uvádzame podrobnejší opis k algoritmu *SPADE*.

* **Algoritmy založené na raste vzorov**. Pomocou známeho prístupu rozdeľuj a panuj je sekvenčná databáza rekurzívne rozdeľovaná na menšie projektované časti, podľa aktuálnych vzorov. Vzory postupne rastú o lokálne frekventované sekvencie hľadané v týchto častiach. Patria tu napr. algoritmy *FreeSpan* (Han, et al., 2000) a *PrefixSpan* (Pei J. , et al., 2001). V prílohe B uvádzame podrobnejší opis k algoritmu *PrefixSpan*.
* **Rozširujúce metódy.** Existuje viacero ďalších metód, ktoré rozširujú klasické prístupy k dolovaniu sekvencií a snažia sa vylepšiť ich problémy s ktorými sa v praxi stýkajú. V krátkosti opisujeme princípy niektorých z týchto zaujímavých algoritmov v časti 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 (pozri časť 0) sú to algoritmy ako *CHARM, CLOSET* , tak v oblasti dolovania frekventovaných sekvencií boli navrhnuté napr. algoritmy *CloSpan* a *BIDE*. Dolovanie uzavretých frekventovaných sekvencií výrazne zmenšuje oblasť prehľadávania a zvýšuje 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 situáciach 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 *približne* 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 aj vzor zhody (*ang. consensus pattern*). Každej sekvencií v zhluku sú pridané tzv. 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 problematické je vhodné nastavenie hladiny minimálnej podpory, pretože ak je príliš vysoká nemusia sa nájsť dôležité vzory a ak je príliš malá môže dôjsť k priveľkému výpočtovému zaťaženiu. Tento problém sa snaží algoritmus *TSP* navrhnutý v práci (Tzvetkov, Yan, & Han, 2005) riešiť nasledovným spôsobom. 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)

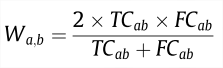
## Iné reprezentácie vzorov správania a prístupy k ich získavaniu

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

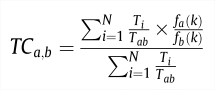
### Algoritmy delenia grafu

V práci (Jalali, Mustapha, Nasir Sulaiman, & Mamat, 2010) sa uvádza zaujímavý prístup hľadania a aplikovania vzorov správania. Z databázy predspracovaných dát zhlukujú používateľské sedenia podľa ich podobných vlastností. Z týchto zhlukov potom vytvára tzv. navigačné vzory, čo je v podstate istá reprezentácia vzorov správania používateľov. Opisovaný proces zhlukovania využíva algoritmus pozostávajúci z týchto základných krokov:

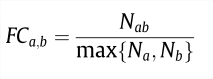
1. Pre každý pár stránok sa počíta stupeň prepojenia na základe vzorca uvažujúceho *frekvencie* spoločného výskytu týchto stránok v používateľských sedeniach a časového rozdielu medzi prístupmi k jednotlivým stránkam v rámci daného sedenia.



* TCa,b je stupeň časového prepojenia medzi návštevami každých dvoch stránok webového sídla.
* FCa,b meria frekvenciu výskytu stránok *a* aj *b* v každom sedení.
* Wa, b stupeň prepojenia medzi stránkami je reprezentovaný ako harmonický priemer časového prepojenia a frekvencie spoločného výskytu stránok v používateľskom sedení.



* Ti je časová dĺžka trvania *i-teho* sedenia, ktoré obsahuje návštevy oboch stránok *a* aj *b.*
* Tab je časový rozdiel medzi časmi navštívenia stránok *a*a *b* v rámci *i*-teho sedenia.
* Jedna z možných reprezentácií funkcie *f* je *fa(k) = k* ak sa stránka *a* nachádza na k-tej pozícií v rámci sedenia. Ak by sme napríklad chceli zvýšiť dôležitosť pozície stránky v rámci sedenia môžeme použiť funkciu *f* kde *f(k) = k2*.



* Nab je počet sedení obsahujúcich stránku *a*aj *b.* Na a podobne aj Nb je počet sedení obsahujúcich len stránku *a*resp. stránku *b.*

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

Získané vzory správania ďalej používajú na generovanie odporúčaní podľa aktuálneho správania používateľov.

### Dolovanie frekventovaných super sekvencií

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.

### Získavanie vzorov správanie pomocou zhlukovania

*Táto časť bude obsahovať jeden dva odseky o práci:*

(Vellingiri, Kaliraj, Satheeshkumar, & Parthiban, 2015)

### SOM

*Táto časť bude obsahovať jeden dva odseky o práci:*

(Etminani, Delui, Yenehsari, & Rouhani, 2009)

## 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 v transakciách 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,* ktorý má najme 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 frekvantované množiny bez potreby generovania kandidátnych frekventovaných množín pomocou kompaktnej dátovej štruktúry s názvom FP-Tree, alebo algoritmus *Eclat,* ktorý zmenšuje nároky na pamäť vďaka rozdeleniu dát na časti, ktoré môžu byť spracované nezávisle. 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 algoritmy *GSP* a *SPADE*. 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 používateľov a prístupov k ich získavaniu napr. pomocou zhlukovania, neurónových sieti či 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 pre veľké a rýchle prúdy dát, čomu sa venujeme v ďalšej kapitole tejto práce.

# Existujúce prístupy hľadania a reprezentácia vzorov správania v prúde dát

V úvode kapitoly 3 sme už uviedli, že v tejto práci budeme pod pojmom vzory správania chápať v literatúre bežne uvádzane tzv. frekventované vzory, čo môžu byť buď frekventované množiny alebo frekventované sekvencie. V tejto časti prenesieme tento problém do oblasti spracovania prúdu dát, kde sa vynárajú nové problémy a výzvy.

V dnešnej dobe v súvislosti s neustálym rastom produkovaných dát pribúdajú aj výzvy na ich efektívne spracovanie. Jedným zo známych vlastností s ktorou sú spájané problémy spracovania veľkých dát je rýchlosť (*ang. Velocity*). Metódy na spracovávanie prúdov dát sa musia vysporiadať s týmto problémom neustáleho rýchleho pribúdania nových inštancií dát. Tie môžu pribúdať teoreticky do nekonečného objemu, ktorý by nebol spracovateľný bežnými metódami určenými pre statické dáta. Získavanie znalostí z prúdov dát je interdisciplinárnou výskumnou oblasťou zaoberajúcou sa metódami a algoritmami získavania znalostí z nestálych dočasných prúdov dát (Krempl, et al., 2014).

Algoritmy spracovania prúdov dát sú stavané na modeloch, ktoré sú inkrementálne aktualizované s prichádzajúcimi inštanciami. Tradičné metódy často vyžadujú viaceré prechody inštanciami v databáze, v prúde dát je každá inštancia spracovávaná práve raz.

Počas vývoja dátového prúdu v čase môže nastávať aj k tzv. konceptuálnemu posunu. Konceptuálny posun je významným problémom v dolovaní prúdov dát a vyplýva zo zmien reálneho prostredia v ktorom dáta vznikajú a jeho nárokov. Faktory vyvolávajúce takéto zmeny sú často skryté a neznáme. Konceptuálny posun spôsobuje vážny problém pri modeloch vytvorených zo starých dát pred týmto posunom, ktoré sa samozrejme v takom prípade správajú 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).

Čo sa týka samotnej úlohy dolovania frekventovaných vzorov v prúde dát, tak existujú tieto tri problémy (Lee, Jin, & Agrawal, 2014):

1. Exponenciálny nárast počtu prehľadávaných vzorov. Základný algoritmus pre statické dáta *Apriori* spracuje podmnožín aby našiel jeden vzor o dĺžke .
2. Pamäťová náročnosť, kvôli veľkému priestoru prehľadávania.
3. Potreba balansovania požiadaviek na presnosť a požiadaviek na efektívnosť. Spresňovanie výsledkov znamená viac zabranej pamäte a viac výpočtového času. Používateľovi by teda malo byť umožnené nastaviť tento balans podľa jeho aktuálnych potrieb.

V časti 4.1 sa budeme venovať podrobnejšie problematike dolovania frekventovaných množín z prúdu dát a bližšie si opíšeme algoritmus *IncMine*, ktorý využívame aj v navrhovanej metóde v kapitole 5. Ďalšie zaujímavé algoritmy ako napr. Closet, Moment opisujeme podrobnejšie v prílohe C.

V časti 4.2 trocha odbočujeme a opisujeme niekoľko možností zhlukovania v prúde dát, keďže významnou súčasťou metódy navrhovanej v kapitole 5 bude aj komponent zabezpečujúci práve tento typ úlohy.

V závere tejto kapitoly (4.3) ešte opisujeme existujúce rámce pre spracovanie prúdu dát, z ktorých jeden používame aj na implementáciu našej metódy.

## Dolovanie frekventovaných množín v prúde dát

V časti 3.1 sme už uviedli úlohu dolovania frekventovaných množín v klasickom statickom prostredí. V tejto časti sa budeme venovať algoritmom, ktorú sa musia vysporiadať s novými požiadavkami, ktoré spracovanie prúdu dát vyžaduje. Rýchlosť pribúdania nových dátových inštancií vylučuje možnosť ukladania celej ich 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 týmito problémami bolo navrhnutých niekoľko prístupov. Väčšinou sú založené na princípe hľadania frekventovaných množíén v rámci okna obsahujúceho posledných *w* transakcií. Toto okno môže byť (Quadrana, Bifet, & Gavaldà, An Efficient Closed Frequent Itemset Miner for the MOA Stream Mining System, 2015):

* **Míľnikové okno** (ang. *landmark window*) , kde predstavuje transakcie v jednej dávke. je najstaršia časť okna a je najnovšia časť.
* **Posúvajúce sa okno** (*ang. sliding window*) , kde *w* je veľkosť posúvajúceho sa okna.
  + S fixovaným časovým intervalom (ang. *time-sensitive*). Počet obsiahnutých transakcií *w* sa rovná počtu transakcií, ktoré prišli v prúde dát za poslednú časovú jednotku .
  + Pevne fixovanej dĺžky (*ang.* *transaction sensitive*), obsahujúce vždy rovnaký počet transakcií.

Prístupy hľadania frekventovaných množín možno ďalej klasifikovať podľa toho koľko transakcií je vstupom do procedúry aktualizácie aktuálnych frekventovaných množín (Quadrana, Bifet, & Gavaldà, An Efficient Closed Frequent Itemset Miner for the MOA Stream Mining System, 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.

Hľadanie exaktných frekventovaných množín v prúde dát môže byť nezvládnuteľné z hľadiska pamäťovej a výpočtovej náročnosti keďže v takom prípade je nutné urdžiavať v pamäti všetky nefrekventované množiny od začiatku behu (Quadrana, Bifet, & Gavaldà, An Efficient Closed Frequent Itemset Miner for the MOA Stream Mining System, 2015).

V tejto kapitole sa bližšie venujeme len algoritmom dolovania uzavretých frekventovaných množín (pozri časť 3.1), keďže sú kompletnou a neredundantnou reprezentáciou všetkých frekventovaných množín a zároveň, ktorá výrazne napomáha k zmenšeniu prehľadávaného priestoru.

### Algoritmy dolovania uzavretých frekventovaných množín v prude dát – porovnanie základných charakteristík

Uvádzame niektoré algoritmy dolovania frekventovaných množín v prúde dát. Existuje viacero algoritmov. My sa zameriavame na algoritmy, ktoré sú určené na dolovanie uzavretých frekventovaných množín, ktoré sú kompletnou a neredundantnou reprezentáciou všetkých frekventovaných množín, ktorá zároveň výrazne napomáha k zmenšeniu prehľadávaného priestoru. Takisto sme sa zamerali na algoritmy využívajúce pohybujúce sa okno, ktoré oproti míľnikovému oknu sa dokáže lepšie vysporiadať s konceptuálnym posunom. (Quadrana & Mestre, 2012)

Prvým navrhnutým algoritmom na inkrementálne dolovanie uzavretých frekventovaných množín MOMENT bol navrhnutý v práci (Chi, Wang, Yu, & Muntz, 2004). Tento algoritmus hľadá exaktné frekventované množiny a využíva pohybujúce sa okno a aktualizáciu v každej transakci. Informácie o všetkých o jednotlivých množinách sa ukladajú do špeciálnej stromovej dátovej štruktúry CET (*ang. closed enumeration tree*), ktorá rozdeľuje do rôznych typov uzlov buď nefrekventované množiny, potenciálne frekventované množiny, a uzavreté frekventované množiny. Na ukladanie informácií o všetkých transakciách v aktuálnom okne využíva dátovú štruktúru *FP-tree*, podobne ako v algoritme *FP-Growth* navrhnutý v práci (Han, Pei, & Yin, 2000) na dolovanie frekventovaných množín v statických datasetochale bez orezávania nefrekventovaných množín. To, že algoritmus MOMENT musí ukladať všetky doteraz nájdené aj nefrekventované množiny aj napriek uvedeným kompaktným štruktúram spôsobuje značné zaťaženie pamäte. Podrobnejšie ho opisujeme v prílohe C. (Quadrana & Mestre, 2012)

Algoritmus NEWMOMENT 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 pôvodnemu algoritmu MOMENT. (Quadrana & Mestre, 2012).

Ďalším algoritmom, ktorý využíva celkom odlišný prístup a iné dátové štruktúry na udržanie informácie o množinách je CLOSTREAM. Podobne ako MOMENT hľadá exaktnú reprezentáciu frekventovaných množín využíva pohybujúce sa okno a aktualizáciu v každej transakci. Podrobnejšie ho opisujeme v prílohe C.

Hľadanie exaktných frekventovaných množín kladie priveľké nároky na algoritmy čo sa týka pamäte. Takéto algoritmy majú tiež problém s adaptáciou v prípade častých výskytov konceptuálnych posunov v prúde dát. Tieto problémy môžu vyriešiť algoritmy, ktoré hľadajú aproximáciu 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 aplikáciach takáto reprezentácia dostatočná a vie sa lepšie vysporiadať aj s konceptuálnym posunom (Quadrana & Mestre, 2012).

IncMine je aproximatívny algoritmus hľadania frekventovaných množín postupne v dávkach aktualizujúci množinu nájdených tzv. uzavretých frekventovaných množín nad pohybujúcim sa oknom v rýchlom prúde dát. Okrem klasickej hladiny minimálnej podpory definuje aj uvoľnenú hladinu minimálnej podpory, ktorá zabezpečuje zotrvanie potenciálne frekventovaných množín v okne. Tento algoritmus bližšie opisujeme v časti 4.1.2.

V práci navrhujú algoritmus s názvom CLAIM. Je to ďalší algoritmus na dolovanie aproximácie uzavretých frekventovaných množín. 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. Aktualizáciu vykonávajú po každej prichádzajúcej transakcií.

### Algoritmus IncMine

IncMine je aproximatívny algoritmus hľadania frekventovaných množín postupne v dávkach aktualizujúci množinu nájdených tzv. uzavretých frekventovaných množín nad rýchlymi prúdmi dát. Tento algoritmus bol navrhnutý v práci (Cheng, Ke, & Ng, 2008).

*Uzavreté frekventované množiny (*ang. *Frequent Closed Itemsets, skr. FCI)* sú frekventované množiny, ktoré nemajú nadmnožinu s rovnakou podporou (napr. frekvencia výskytov) . Dôležitou vlastnosťou množiny všetkých FCIje, že je kompletná a neredundantná reprezentácia všetkých frekventovaných množín a často omnoho menšia ako množina všetkých frekventovaných množín, tak ako sme si to uviedli v úvode tejto kapitoly.

Nech je databáza transakcií. Nech je množina všetkých položiek v databáze. Transakčný dátový prúd je sekvencia po sebe prichádzajúcich transakcií. Transakcia je teda množinou položiek. Platí Y. Nech *X*  je množinou X. Časová jednotka nech je označená ako . V časovej jednotke môže prísť v prúde variabilný počet transakcií. Okno *W* je časovým intervalom – množina po sebe nasledujúcich časových jednotiek *.* Posúvajúce sa okno *W* po prúde je okno s fixným počtom časových jednotiek, ktoré sa posúva dopredu po každej časovej jednotke. Nech je aktuálna časová jednotka a veľkosť okna je *w* – potom je aktuálne okno.

Nech je množina transakcií, ktoré prišli v prúde v rámci aktuálneho okna *w.* Nech *support(X,* *)* označuje počet transakcií Y z v ktorých sa nachádza *X.*

Nech označuje minimálnu hranicu podpory. Ak tak *X* prehlásime za frekventovanú množinu. Okrem toho *X* je uzavretá frekventovaná množina (*FCI*) nad ak platí : X je frekventovaná množina nad a

Riešime problém hľadania množiny všetkých *FCI* v rámci posúvajúceho sa okna *W.* Problémom je zabezpečiť zotrvanie nefrekventovaných množín s potenciálom stať sa frekventovanými počas behu algoritmu. Predpokladajme, že množina *X*, ktorá nie je frekventovaná sa môže po čase *t* stať frekventovanou. Posúvaním okna sa však podpora množiny *X*, ktorú mala pred časom *t*, stratí*.* Napr. v práci (Chang & Lee, 2003) tento problém riešia pomocou tzv. *uvoľnenej hranice minimálnej podpory* (ang. *relaxed minimum support threshold*). Množina *X* zotrvá v okne pokiaľ jej podpora je väčšia ako . Čím menšie je tým viac sa blíži podpora množiny *X* jej skutočnej podpore avšak tým väčšie je aj množstvo uchovávaných množín. V  práci (Cheng, Ke, & Ng, 2008) preto navrhli pojem tzv. *semi-frekventovaných množín*, kde definujú , kde ) je tzv. *rýchlosť uvoľňovania* . Definujú progresívne rastúcu funkciu minimálnej podpory *minsup(k)* ako

)

*minsup(k)* počítame pre všetky *k* , . Kde . Aproximatívnu hodnotu podpory *X* nad časovou jednotou *t* vypočítame ako (kde je množina transakcií v časovej jednotke *t*):

Aproximatívnu podporu nad časovým intervalom vypočítame ako :

*X* je semi-frekventovaná uzavretá množina (*semi-FCI*) nad oknom *W* ak :

)

*X* je tzv. *k – semi –frekventovaná množina (k-semi-FCI)* nad oknom W ak je to *semi-FCI* a *k* je dané ako najväčšie *k*pre ktoré platí : .

Uvedená funkcia *minsup* nemusí byť vhodná pre všetky druhy dátových prúdov. Štúdiom charakteristík dátového prúdu môžeme túto funkciu prispôsobiť aby čo najlepšie zachytávala zmeny v dátových prúdoch.

Nech C je množina *semi-FCI* nad aktuálnym oknom, ktorú chceme získať. L je množina *semi-FCI* nad predchádzajúcim oknom. A F je množina *semi-FCI* z aktuálnej časovej jednotky . Úlohu inkrementálnej aktualizácie množiny *semi-FCIs* nad pohybujúcim sa oknom definujú autori ako získanie množiny C aktualizovaním množiny L pomocou F. Výstupom je nakoniec množina všetkých *k-semi-FCI*, ktoré majú vyššiu podporu ako je minimálna podpora pre okno Wc, teda .

Autori ďalej skúmali vlastnosti *semi-FCI* a uvádzajú niekoľko tvrdení vďaka, ktorým ich algoritmus *IncMine* inkrementálne aktualizuje množiny semi-FCIs efektívnejšie ako jeho predchodcovia. Ďalej v opisovanej práci autori uvádzajú, že klasická dátová štruktúra *prefix-tree*, ktorá sa používa pri dolovaní frekventovaných vzorov nie je dostatočne efektívna hlavne pre operáciu hľadania *semi-FCI* a preto navrhujú použitie *invertovaného indexu* ako dátovej štruktúry na ukladanie množín *F*, *C* a *L*. Nakoniec v práci experimentálne potvrdzujú efektivitu a rýchlosť algoritmu ako i porovnateľnú presnosť a pokrytie v porovnaní s inými algoritmami.

## Zhlukovanie nad prúdom dát

Zhlukovanie nad prúdom dát je náročnou úlohou. V tejto podkapitole analyzujeme niekoľko základných princípov a algoritmov zhlukovania nad prúdom dát. Tieto znalosti ďalej využívame v návrhu metódy na zhlukovanie modelov používateľov v kapitole 5. Barbará vo svojej práci (Barbará, 2003) hovorí o všeobecných požiadavkách na algoritmy zhlukovania nad prúdom dát:

* Kompaktnosť reprezentácie zhlukov. Teda dôležité je aby veľkosť reprezentácie nerástla rýchlo zároveň s prichádzajúcimi novými dátovými inštanciami.
* Rýchle a inkrementálne spracovanie nových dátových inštancií.
* Jasné a rýchle identifikovanie odľahlých dátových inštancií.

Odteraz pre zjednodušenie pokiaľ hovoríme o zhlukovaní tak myslíme zhlukovanie nad prúdom dát. Bolo navrhnutých viacero algoritmov napr. Clustream v práci (Aggarwal, a iní, 2003). It is based on online microclustering component performing fast transformation of incoming data instances into compact approximate statistical representation and offline component using this representation to perform macroclustering and get results of clustering on demand. This approach is adapted in other works using different macroclustering algorithms and altering microclustering phase slightly, like density based algorithm Denstream proposed in [12] that uses DBScan as macroclustering algorithm and defines new concepts of core-microclusters and outlier-microclusters. Denstream is able to detect clusters of arbitrary shapes and requires no assumption on the number of clusters. CluStream approach is adapted also in HPStream – projected clustering for high-dimensional data streams proposed in [13]. It outperforms basic CluStream with high-dimensional streaming data. There are also other algorithms not based on CluStream. For example another density based clustering algorithm D-Stream proposed in [14].

HPStream (Aggarwal, Han, Wang, & Yu, 2004), DenStream (Cao, Ester, Qian, & Zhou, 2006), Dstream (Chen & Tu, 2007), ClusTree (Kranen, Assent, Baldauf, & Seidl, 2011). Dôležitý rámec pre zhlukovanie v prúdoch dát Clustream, ktorý je základom pre niektoré ďalšie algoritmy opisujeme v časti 4.2.1. Niektoré ďalšie ako DenStream a ClusTree opisujeme v prílohe D.

In our method we decided to use CluStream with k-means macroclustering algorithm. As we want our method to perform macroclustering regularly as part of data streaming processing, we want this phase to be fast enough. K- means is fast and easy to adjust algorithm which helps us quickly prove concepts we define in this work. But the process we define in this work should be considered as framework and macroclustering algorithm as well as frequent patterns mining algorithm should be matter of choice depending on usage domain requirements

### Rámec Clustream

V práci (Aggarwal, a iní, 2003) navrhli rámec s názvom Clustream na zhlukovanie nad prúdom dát, ktorý je založený na dvoch komponentoch. Online mikro-zhlukovací komponent, zabezpečujúci rýchle spracovanie prichádzajúcich údajov a efektívne ukladanie sumárnych štatistík. A offline komponent používajúci tieto sumárne štatistiky spoločne s ďalším používateľským vstupom, ktorý používateľovi ukazuje informácie o existujúcich zhlukoch na požiadanie.

Online komponent navrhnutého rámca v každom čase vykonávania algoritmu udržuje štatistiky o stave *q* mikro-zhlukov. Označme tieto mikrozhluky ako . Počet mikrozhlukov *q* je určený veľkosťou dostupnej hlavnej pamäte. Typicky je hodnota *q* značne vyššia ako počet hľadaných zhlukov ale tiež značne menšia ako počet analyzovaných dátových inštancií. Mikrozhluky predstavujú snímku aktuálneho stavu zhlukov nachádzajúcich sa v dátovom prúde. História vývoja mikrozhlukov sa neukladá. Na začiatku algoritmu sa offline vytvoria pomocou klasického k-means zhlukovania z určených počiatočných *N* dátových bodov iniciálne mikrozhluky. Po tejto inicializácií algoritmus pokračuje v online fáze postupnej aktualizácie mikrozhlukov. Ku aktualizácií dochádza vždy s príchodom novej dátovej inštancie . Určí sa vzdialenosť dátovej inštancie ku všetkým centroidom existujúcich mikrozhlukov. Nech najbližší nájdený centroid je označený ako Môžu nastať nasledovné situácie:

* Dátová inštancia je pohltená mikrozhlukom.
* Z dátovej inštancie je vytvorený nový mikrozhluk.

Na určenie toho či má byť dátová inštancia pohltená mikrozhlukom je nutné určiť tzv. maximálnu hranicu mikrozhluku, ktorá je definovaná ako stredný štvorcový rozptyl (ang. Root Mean Square Deviation, skr. RMSD) vzdialeností dátových inštancií patriacich konkrétnemu mikrozhluku. Ak je vzdialenosť dátovej inštancia k centroidu najbližšieho mikrozhluku nižšia ako jeho maximálna hranica tak je jednoducho pohltená mikrozhlukom a prepočíta sa jeho maximálna hranica. Ak sa dátová inštancia nevošla do maximálnej hranice je z nej vytvorený nový mikrozhluk. Iniciálna maximálna hranica mikrozhluku je určená heuristicky. Ak sa však takýmto spôsobom vytvorí nový mikrozhluk tak je potrebné iný mikrozhluk zrušiť alebo dva mikrozhluky spojiť, aby sa zachoval počet *q*. Prvým krokom je teda hľadanie starého mikrozhluku, ktorý leží príliš mimo ostatných mikrozhlukov (v ang. outlier). Ak je takýto identifikovaný tak je bezpečné ho odstrániť. Autori rámca navrhli identifikovať takéto mikrozhluky pomocou približnej priemernej časovej pečiatky posledných m pridaných dátových inštancií. Ten mikrozhluk, ktorý má túto priemernú časovú počiatku najstaršiu a zároveň staršiu ako je používateľom stanovená hranica môže byť odstránený. V prípade, že žiaden z mikrozhlukov nemôže byť odstránený, musí dôjsť k spojeniu dvojice mikrozhlukov s najmenšiou vzájomnou vzdialenosťou.

Offline komponent zabezpečuje používateľovi možnosť skúmať zhluky v dátovom prúde v rôznych časových momentoch. Vďaka štatistickej reprezentácií dát pomocou mikrozhlukov je offline proces hľadania makrozhlukov značne zrýchlený. Makrozhluky sú získavané pomocou modifikovaného k-means algoritmu. Mikrozhluky sú chápané ako zhlukované dátové inštancie. Vo fáze inicializácie sa nevyberajú počiatočné body náhodne ale mikrozhluky s vyšším počtom bodov sú vybrané pravdepodobnejšie.

## Existujúce rámce na spracovávanie kontinuálneho prúdu dát

V tejto časti analyzujeme niekoľko známych existujúcich rámcov určených pre prácu a dolovanie znalosti z prúdu dát.

### RAPID MINER – Streams plugin

RapidMiner (skr. RM) je softvérová platforma určená na rýchly návrh aj zložitých problémov strojového učenia a dolovania znalostí v dátach, ktoré sa modelujú z procesného pohľadu. Na opis procesov používa zreťazené štruktúry operátorov definovaných v XML. RM je implementovaný v Jave a je Open Source. Samotný RM je možné použiť iba v prípade učenia v dávkach. Na spracovanie kontinuálneho prúdu dát slúži doplnková knižnica *streams* integrovateľná do RM ako StreamsPlugin. Knižnica streams poskytuje triedy a rozhrania, ktoré zabezpečujú spracovanie prúdov dát z rôznych zdrojov. Neponúka však implementácie algoritmov strojového učenia pre prúdové dáta. Je však možné integrovať do procesu spracovania algoritmy dostupné napríklad z rámca MOA (Bockermann & Blom, 2012).

### MOA – Massive Online Analysis

*MOA* je softvérový rámec implementovaný v jazyku Java. Obsahuje kolekciu vopred implementovaných a umožňuje vytváranie nových algoritmov pracujúcich s masívnymi vyvíjajúcimi sa prúdmi dát a poskytuje aj možnosti na ich evaluáciu. Je voľne dostupná pod *GNU PL* licenciou a podporuje interakciu s knižnicou *WEKA (Waikato Environment for Knowledge Analysis)* tiež implementovanou v jazyku Java a obsahujúcou širokú škálu implementovaných dávkových metód strojového učenia. (Bifet, Holmes, Kirkby, & Pfahringer, 2010)

Oproti bežnému dávkovému spracovaniu je odlišný aj spôsob evaluácie prúdových dát. Dva známe prístupy, ktoré sú implementované aj v rámci *MOA* sú (Bifet, Holmes, Kirkby, & Pfahringer, 2010):

* Otestovanie modelu na vopred určenej nazbieranej podmnožine dát (v podstate rovnako ako pri rozdelení na trénovaciu a testovaciu množinu pri dávkových metodách strojového učenia)
* Testovanie prepletené s trénovaním. Každá vzorka je použitá na otestovanie modelu predtým než sa použije na jeho trénovanie. Z výsledkov takýchto individuálnych testov sa inkrementálne aktualizujú metriky ohodnocujúce model (napr. presnosť klasifikácie).

### SAMOA - Scalable Advanced Massive Online Analysis

*SAMOA* je platforma pre dolovanie v masívnych dátových prúdoch. Podobne ako *MOA* obsahuje kolekciu state-of-art algoritmov pracujúcich s masívnymi a vyvíjajúcimi sa prúdmi dát a tiež programové abstrakcie umožňujúce implementáciu nových algoritmov. Rozhodujúcou výhodou oproti *MOA* je, že umožňuje tieto algoritmy vykonávať vo viacerých  rámcoch umožňujúcich distribuované spracovanie ako *Apache STORM*, *S4* a *SAMZA*. *SAMOA* je implementovaná v Jave a je Open Source (Morales & Bifet, 2015).

## Sumarizácia a diskusia

V úvode tejto kapitoly sme opísali nové požiadavky, ktoré sú kladené na úlohu hľadania frekventovaných vzorov v prúde dát. Hovorili sme o požiadavkach na rýchlosť, pamäťovú efektívnosť a požiadavke na možnosť balansovať efektívnosť výpočtu s presnosťou 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. Ďalej možno prístupy deliť podľa toho či aktualizujú frekventované množiny v dávkach alebo s každou prichádzajúcou transakciou, či hľadajú exaktné frekventované množiny, alebo len aproximatívne frekventované množiny, čo vedie k rýchlejšiemu a pamäťovo efektívnejšiemu spracovaniu. Bližšie sme analyzovali algoritmy *Moment*, *Closet* a *IncMine* a uviedli ich krátke porovnanie. Uviedli sme podrobnejší opis fungovania algoritmu *IncMine.* Opisy algoritmov *Moment* a *Closet* uvádzame pre ich rozsiahlosť v prílohe C.

Algoritmus *IncMine* s niektorými úpravami sme sa rozhodli použiť na implementáciu komponentu metódy, ktorú navrhujeme v kapitole 5 zabezpečujúceho dolovanie frekventovanýh množín z prúdu dát aj z nasledujúcich dôvodov:

* Jeho implementácia je dostupná v rámci knižnice MOA.
* Je aproximatívny algoritmus s veľmi dobrou presnosťou aproximácie podľa výsledkov experimentov uvedených v práci, čo prispieva k vyššej rýchlosti spracovania oproti exaktným algoritmom ako *MOMENT* a *CLOSTREAM*.
* Hľadá uzavreté frekventované množiny, ktoré sú kompletnou a neredundantnou reprezentáciou všetkých frekventovaných množín, ktorá zároveň výrazne napomáha k zmenšeniu prehľadávaného priestoru.
* využíva pohybujúce sa okno, ktoré oproti míľnikovému oknu sa dokáže lepšie vysporiadať s konceptuálnym posunom.
* Autori sa  porovnávajú s algoritmom *MOMENT*. *IncMine* je podľa ich výsledkov značne rýchlejší.
* *IncMine* ako jediný z analyzovaných algoritmov aktualizuje frekventované množiny po dávkach transakcií ako autori pozorujú vo svojich experimentoch zdá sa tento prístup vhodnejší keďže v prípade aktualizácie po každej transakcií je aktualizácie príliš častá a môže viesť k zbytočne častému vyťaženiu výpočtových zdrojov, prínos jednej jedinej transakcie oproti väčšej dávke transakcií spolu je pritom často zanedbateľný.

Opísali sme aj problém zhlukovania v prúde dát a algoritmy *Clustream* a *Clustree* sme bližšie opísali v prílohe D. Tomuto problému sme sa venovali kvôli tomu, že významnou súčasťou metódy navrhovanej v kapitole 5 bude práve komponent zabezpečujúci zhlukovanie modelov používateľov na základe ich správania. Rozhodli sme sa využiť *CluStream* ako základný algoritmus pre zhlukovanie modelov používateľov. Ako sme spomenuli tento algoritmus predstavuje rámec na ktorom sú postavené aj ďalšie algoritmy zhlukovania v prúde dát (napr. *DenStream*). Proces navrhovaný v kapitole 5 sme sa snažili navrhnúť ako rámcový process v ktorom je možné neskôr *CluStream* zameniť za iný algoritmus založený na podobnom princípe dvojfázov.

# Navrhovaná metóda reprezentácie, extrakcie a aplikácie vzorov správania nad prúdom dát

V tejto časti opisujeme návrh metódy, ktorá zabezpečuje jednak hľadanie *globálnych vzorov* správania typických pre všetkých používateľov a zároveň segmentáciu používateľov do skupín a hľadanie *skupinových vzorov* správania, teda vzorov typických pre užšie skupiny používateľov. V návrhu tiež opisujeme spôsob aplikácie kombinácie globálnych a skupinových vzorov správania v úlohe odporúčania. Neskôr v kapitole 7 skúmame či vďaka navrhovanej metóde a kombinácií globálnych vzorov so  skupinovými vieme dosiahnuť lepšie výsledky v úlohe odporúčania než by bolo možné dosiahnuť len s použitím globálnych vzorov.

Dôležitou charakteristikou metódy je práca s prúdom dát. Ako sme uviedli v úvode kapitoly 4 kladie to na navrhovanú metódu isté požiadavky s ktorými sme pri návrhu museli rátať. Naša metóda je novou aplikáciou v oblasti dolovania dát o používaní Webu, ktorá v porovnaní voči aplikáciam, ktoré sme uviedli v časti 0 reaguje na niektoré ďalšie výzvy a to konkrétne personalizáciu Webu (keďže hľadá vzory typické k správaniu používateľa) a spracovanie dát ako rýchleho a potencionálne nekonečného prúdu. Prínos navrhovanej metódy spočíva v schopnosti 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. Nachádzané vzory správania majú viacero možných aplikácií (pozri časť 0) ako napr. použitie na predikciu, odporúčanie, prednačítavanie stránok do dočasnej pamäte, či celkové porozumenie správania používateľov webového sídla.

Návrh pozostáva z častí venujúcich sa formátu spracovávaných dát (5.1.1), spôsobu reprezentovania vzorov správania (5.1.2) , modelu používateľa (5.1.3), spôsobu aplikácie vzorov správania v úlohe odporúčania (5.1.4), rámcového opisu procesu spracovania transakcií (0). rád pre realizáciu v praxi (5.2.1), návrhu paralelizácie procesu (5.2.2), spôsobu regulovania rýchlosti spracovania transakcií (5.2.3). Na záver ešte sumarizujeme všetky vstupné parametre metódy (0).

## Základné pojmy

Táto časť objasňuje niektoré základné pojmy a princípy navrhovanej metódy.

### 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 usporiadaný zoznam akcií používateľa v rámci sedenia. Za akcie sa môžu považovať napríklad návštevy konkrétnych stránok alebo širších kategórií stránok, ale môžu to byť aj iné akcie používateľa v systéme ako napr. pridanie položky do košíka v internetovom obchode, vymazanie položky z košíka, pridanie komentára, pridanie hodnotenia k položke, kúpa a mnoho ďalších. Okrem toho obsahuje transakcia identifikátor používateľa, ktorý dané akcie vykonal.

### Reprezentácia vzoru správania

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 group identificator*).
* Uzavretú frekventovanú množinu (pozri 3.1) (*ozn.* *p.fci z ang. pattern frequent closed itemset*). To je množina identifikátorov akcií používateľa, ktorá spĺňa podmienku minimálnej podpory nad aktuálnym pohybujúcim sa oknom (pozri časť 4.1.2) v rámci prúdu dát.
* Váhu vzoru (*ozn. p.sup z ang. pattern support*). To je hodnota podpory množiny vypočítaná vzhľadom k aktuálnemu pohybujúcemu sa oknu v prúde dát. Čo je podpora množiny sme vysvetlili v časti 3.1.

### Reprezentácia modelov používateľov

Keďže jedným z cieľov navrhovanej metódy je hľadanie skupinových vzorov správania v prúde dát, je dôležitým problémom, ktorý musí metóda riešiť aj reprezentácia modelov použivateľov s čo najmenšími nárokmi na pamäť a tiež zabezpečenie aby využitie pamäte na uloženie týchto modelov nerástlo proporcionálne k nárastu 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*).
* Identifikátor posledného makrozhlukovania v ktorom bol používateľovi priradený skupinový identifikátor (*ozn. u.lmid z ang. user’s last macroclustering identificator*).

Vo svete webových sídel môže byť počet používateľov nesmierne dynamický. Noví používatelia prichádzajú a starí odchádzajú. Navrhovaná metóda v pravidelných intervaloch aktualizuje zhluky používateľov 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 založeného na rámci *CluStream* (ktorý sme opísali v časti 4.2.1) je nová dátová inštancia vygenerovaná na požiadanie z histórie uskutočnených akcií modelu používateľa v podobe zoznamu frekvencií jednotlivých vykonaných akcií. Každé nové uskutočnené makrozhlukovanie má priradené nové identifikačné číslo a vygeneruje nové zhluky s novými centrami pravidelne po určenom počte aktualizácií mikrozhlukov (bližšie opisujeme tento proces ďalej v časti 5.2). Priradenie identifikátoru skupiny používateľovi sa deje vždy len pri prvej vyskytnutej požiadavke (teda prvom sedení používateľa od posledného makrozhlukovania) teda iba vtedy ak je identifikátor posledného makrozhlukovania v modeli používateľa *u* (atribút *u.lmid* ) iný než identifikátor najnovšieho makrozhlukovania.

Aby sa proporcionálne k času behu metódy nezvyšovalo zaťaženie pamäte tým, že si bude pamätať všetkých používateľov od začiatku behu tak sa pravidelne zoznam modelov používateľov prechádza a odstráni sa každý taký model používateľa *u* pre ktorý platí, že *lmid* - *u.lmid > tmdiff* , kde *lmid* je identifikátor najnovšieho makrozhlukovania a *tmdiff* je vstupný parameter (sumarizácia parametrov je v časti 5.3).

Dôležitým problémom pri algoritmoch strojového učenia býva tiež dimenzionalita dát. S narastajúcou dimenzionalitou rôznych akcií sa bude spomaľovať aj algoritmus pre zhlukovanie ale aj algoritmus hľadania uzavretých frekventovaných množín, ktoré sú súčasťou navrhovanej metódy. Preto je dôležité pri generovaní dát, ktoré sú vstupom metódy premýšlať aj s ohľadom na toto zaťaženie a pokiaľ sa dá tak generovať dátové inštancie s menšou veľkosťou množiny možných akcií používateľa (dimenzionalitou).

### Spôsob aplikácie nájdených vzorov

Výstupom navrhnutej metódy je v každom čase množina objavených globálnych vzorov správania a množina objavených skupinových vzorov správania. Úlohu hľadania vzorov správania môžeme považovať za samostatnú a oddelenú od úlohy aplikácie vzorov správania. Tieto úlohy teda samozrejme môžu byť realizované aj rôznymi fyzickými komponentami architektúry konkrétneho softvéroveho riešenia. Vygenerované vzory správania môžu byť použité na rôzne ciele ako napr. generovanie odporúčaní , analýzu správania používateľov za rôznymi účelmi ako napríklad zlepšenie štruktúry a dizajnu webového sídla, pochopenia používateľa a jeho chovania ako sme to uviedli v časti 2.2.

My sme sa rozhodli skúmať vzťahy medzi skupinovými a globálnymi vzormi správania (okrem iných skúmaných štatistík) najmä prostredníctom aplikácie týchto vzorov v úlohe odporúčania a skúmania prínosu skupinových vzorov v kombinácií s globálnymi práve v tejto úlohe.

Dôležitým problémom pri generovaní odporúčaní je vybranie vzorov správania, ktoré čo najlepšie vystihujú súčasné správanie používateľa. Nech aktuálne sedenie je reprezentované vektorom akcií používateľa a tiež všetky vzory správania sú reprezentované ako vektory položiek. Vstupným parametrom metódy je veľkosť okna vyhodnocovania *ews* (skr. z ang. *evaluation window size*). Platí . Aktuálne sedenie *S* sa rozdelí na časť, ktorá sa použije na hľadanie vzoru a testovaciu časť, ktorá sa použije na vyhodnotenie odporúčania. Označme časť určenú na hľadanie vzoru ako vyhodnocovacie okno . A označme časť určenú na testovanie ako Nech počet odporúčaných položiek je *rc* (to je vstupný parameter metódy, pozri 5.3). Ak aktuálne sedenie nespĺňa podmienku *n >= (ews + rc)* tak ho pre účely vyhodnocovania odporúčania ignorujeme. V našej metóde používame nasledovnú stratégiu výberu vzorov správania:

1. Pre každý vzor správania vypočítame odhadovanú podporu normalizovanú na interval <0,1> a tiež nájdeme prienik s aktuálnym vyhodnocovacím oknom sedenia používateľa pomocou algoritmu *LCS* (*Longest common subset*). Označme hodnotu tohto prieniku (teda počet spoločných akcií normalizovaný na hodnotu z intervalu <0,1> podľa dĺžky okna ) ako .
2. Všetky nájdené vzory sa zoradia.
   1. Prvá úroveň zoradenia je podľa hodnoty prieniku od najväčšieho po najmenšiu hodnotu.
   2. Druhá úroveň zoradenia je podľa hodnoty podpory vzoru správania od najvyššej po najmenšiu.
3. Postupne sa striedavo prechádzajú globálne a skupinové vzory od najlepšieho k horším a buduje sa mapa kde sa jednotlivým položkám, ktoré sa nachádzajú v týchto vzoroch (a nenachádzajú sa vo vyhodnocovaciom okne ) pridávajú „hlasy“. Hodnota položky sa inkrementuje o hodnotu podpory vzoru v ktorom sa nachádza násobenej hodnotou prieniku vzoru s aktuálnym oknom sedenia teda (obidve normalizované do intervalu <0,1>):

Toto sa opakuje pre každý nájdený vzor. Nakoniec sa vyberie *rc* (vstupný parameter metódy udávajúci počet odporúčaných položiek) položiek s najvyššou hodnotou hlasov a tie sa odporúčia používateľovi.

## Proces spracovania transakcií

V tejto časti opisujeme proces spracovania vstupných transakcií prúdiacich do metódy. Ide o návrh platný pre experimentálne prostredie. To znamená že proces je nastavený tak aby zahŕňal aj kroky potrebné pre ohodnotenie úspešnosti metódy.

V časti 5.2.1 opisujeme aké zmeny v tomto procese by bolo vhodné urobiť v prípade zavedenia metódy do prostredia reálnej aplikácie.

Navrhovaný proces predstavuje rámcové riešenie problematiky dolovania skupinových vzorov z prúdu dát keďže niektoré komponenty možno zamieňať (zhlukovací algoritmus, algoritmus dolovania uzavretých frekventovaných množín).

Celý proces spracovania jednej transakcie reprezentujúcej sedenie používateľa je znázornený na diagrame aktivít (Obrázok 1). V tejto časti bližšie opíšeme jednotlivé kroky. V diagrame identifikátory začínajúce písmenom A označujú bloky aktivít, začínajúce písmenom D dátové úložiská (v pamäti alebo externé), začínajúce písmenom O dátové objekty.

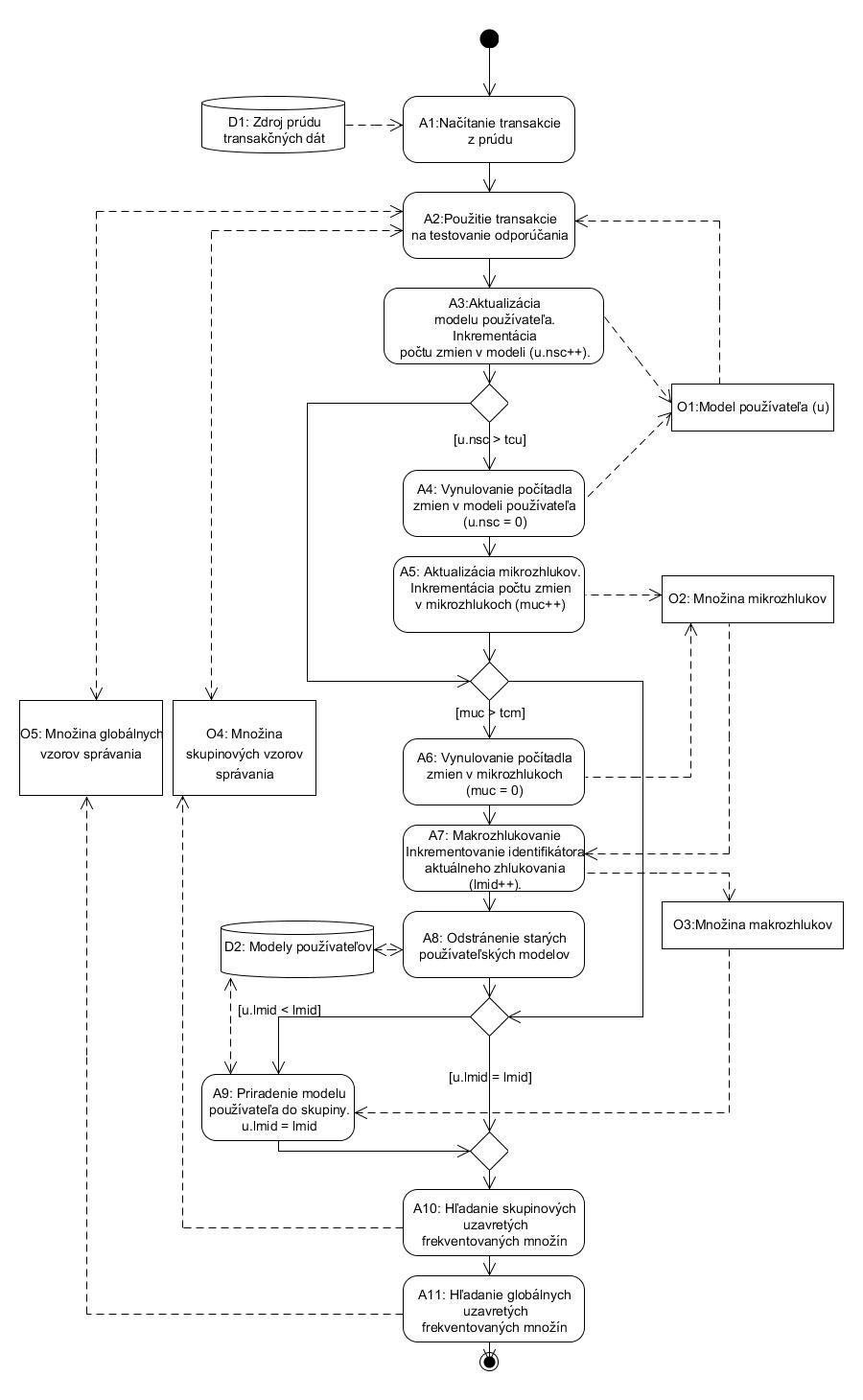
* **D1: Zdroj prúdu transakčných dát.** V tejto práci používame kvôli zjednodušeniu vyhodnocovania vopred pripravené dáta v požadovanom formáte. V reálnej aplikácií by boli tieto dáta generované webovým sídlom samotným. Formát dát sme opísali v časti 5.1.1.
* **D2: Modely používateľov.** V pamäti sa udržiava množina všetkých modelov aktívnych používateľov, ktoré sú neskôr vstupom pre zhlukovanie. O použití a obmedzeniach veľkosti tejto množiny a spôsobe reprezentácie modelov používateľov bližšie píšeme v časti 5.1.3.
* **O1:** **Model používateľa.** Tento objekt reprezentuje konkrétny model používateľa. Reprezentácia modelu používateľa je bližšie opísaná v časti 5.1.3.
* **O2:Množina mikrozhlukov.** Objekt reprezentuje množinu mikrozhlukov udržiavaných v pamäti. Mikrozhluky predstavujú štatistickú snímku aktuálneho stavu zhlukov nachádzajúcich sa v dátovom prúde. Viac o mikrozhlukoch v časti 4.2.1 venujúcej sa rámcu *CluStream* určenému na zhlukovanie nad prúdom dát, ktorý sme sa rozhodli použiť.
* **O3:Množina makrozhlukov:** Objekt reprezentuje množinu makrozhlukov udržiavaných v pamäti. Makrozhluky predstavujú aproximáciu aktuálneho stavu zhlukov v prúde dát, ktorý je získaný pomocou upraveného algoritmu *k-means* z množiny mikrozhlukov. Viac o makrozhlukoch v časti 4.2.1 venujúcej sa rámcu *CluStream* určenom na zhlukovanie nad prúdom dát, ktorý sme sa rozhodli použiť.
* **O4: Množina skupinových vzorov správania:** Objekt reprezentuje množinu aktuálnych objavených skupinových vzorov správania.
* **O5: Množina globálnych vzorov správania:** Objekt reprezentuje množinu aktuálnych objavených globálnych vzorov správania.
* **A1: Načítanie transakcie z prúdu dát.** Prvým krokom procesu spracovania transakcie reprezentujúcej sedenie používateľa je jej načítanie zo zdroja prúdu dát **D1**.
* **A2: Použitie transakcie na testovanie odporúčania.** Prvým krokom, ktorý s transakciou vykonáme je otestovanie na úlohu odporúčania. Spôsob odporúčania opisujeme v časti 5.1.4. Spôsob samotného vyhodnocovania odporúčania pomocou konkrétnych metrík opisujeme v časti 7.1.1.
* **A3: Aktualizácia modelu používateľa. Inkrementácia počtu zmien v modeli.** Nasleduje aktualizácia modelu používateľa *u* (**O1**) podľa údajov aktuálneho sedenia.

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).

* **A4: Vynulovanie počítadla zmien v modeli používateľa.** Ak počítadlo zmien v modeli používateľa *u* (atribút *u.nsc*) ešte nedosiahlo hodnotu vyššiu ako vopred definovaná hraničná hodnota počtu zmien v modeli používateľa *(skr. tcu z ang. threshold number of changes in usermodel)* tak sa kroky **A4** aj **A5** preskočia.
* **A5: Aktualizácia mikrozhlukov. Inkrementácia počtu zmien v mikrozhlukoch.**

V prípade, že došlo k dostatočnému počtu zmien v modeli používateľa *u* (atribút *u.nsc*) sa tieto zmeny prenesú aj do do štatistickej reprezentácie zhlukov – tzv. mikrozhlukov. Na zhlukovanie v prúde dát používame rámec *CluStream*. Ten sme už opísali v časti 4.2.1. Počítadlo zmien v mikrozhlukoch *muc* (*skr. z ang. microclusters updates counter*) je inkrementované s každým vykonaním tohto kroku **A5** o 1.

* **A6: Vynulovanie počítadla zmien v modeli používateľa.** Ak počítadlo zmien v mikrozhlukoch *muc* ešte nedosiahlo hodnotu vyššiu ako vopred definovaný parameter*tcm* (*hraničný počet zmien v mikrozhlukoch skr. z ang. threshold of changes in microclusters*) tak sa krok **A6** aj **A7** preskočia.
* **A7: Makrozhlukovanie.** Ak došlo k dostatočnému počtu zmien v mikrozhlukoch tak sa z tejto štatistickej reprezentácie pomocou klasického zhlukovacieho algoritmu získajú tzv. makrozhluky. Tu možno použiť modifikovaný algoritmus *k-means*, ktorý hľadá vstupným parametrom *gc* (*z ang. groups count*) určený počet makrozhlukov zo vstupných mikrozhlukov. Alebo napr. v prípade použitia algoritmu *DenStream* (opísaný v prílohe D) sa použije ako makrozhlukovací algoritmus *DBSCAN,* ktorý nevyžaduje vopred určiť počet makrozhlukov. Inkrementuje sa tiež identifikátor posledného vykonaného makrozhlukovania *lmid*. Viac o spôsobe zhlukovania v prúde dát v časti 4.2.1.
* **A8: Odstránenie starých modelov používateľov.** Tento krok zabezpečuje, že pri každom novom makrozhlukovaní sa tiež zabezpečí odstránenie starých modelov používateľov reprezentujúcich používateľov, ktorý už dlhšie neboli aktívni. Odstránia sa tie modely, pre ktoré platí, že rozdiel medzi identifikátorom aktuálneho makrozhlukovania *lmid* a identifikátorom posledného markozhlukovania v modeli používateľa *u* (atribút *u.lcid* ) je väčší ako hodnota vstupného parametra *tcdiff* (*hraničná hodnota rozdielu identifikátorov zhlukovania skr. tcdiff z ang. threshold of clustering id’s difference*). Teda ak platí: *lmid – u.lmid > tdiff* .
* **A9:** **Aktualizácia priradenia modelu používateľa do skupiny.** Ak sa nerovnajú identifikátory zhlukovania v modeli používateľa *u* a aktuálneho makrozhlukovania (*u.lmid < lmid*) tak sa *u.lmid* modelu používateľa nastaví na rovnakú hodnotu ako *lmid* a modelu používateľa sa priradí identifikátor priradenej skupiny (atribút *u.gid*).
* **A10:** **Hľadanie skupinových uzavretých frekventovaných množín.** Ak model používateľa *u* už má priradený identifikátor skupiny (atribút *u.gid*) tak je aktuálne sedenie vstupom do algoritmu hľadania uzavretých frekventovaných množín *IncMine*, ktorý sme opísali v časti 4.1.2. 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. A uzavreté frekventované množiny nájdené pre konkrétnu skupinu sú uložené do samostatnej dátovej štruktúry **O4**.
* **A11:** **Hľadanie globálnych uzavretých frekventovaných množín.** 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*, ktorý sme opísali v časti 4.1.2. Nájdené uzavreté frekventované množiny sa ukladajú do samostatnej dátovej štruktúry určenej pre globálne uzavreté frekventované množiny **O5**.



Obrázok 1 Proces spracovania transakcie v navrhovanej metóde .

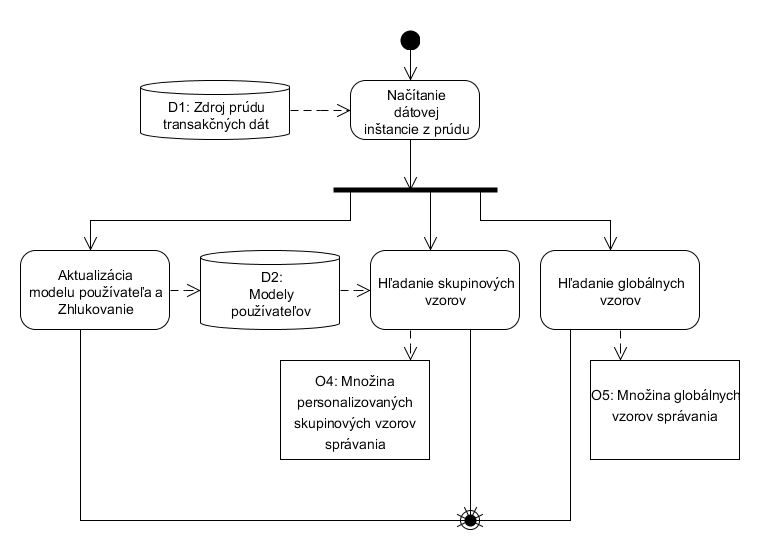
### Zmeny v procese v prípade reálnej aplikácie

Proces, ktorý sme opísali je navrhnutý tak ako sme ho použili pre vykonanie experimentov opísaných aj v kapitole 7. V prípade, kedy by sa mala metóda použiť v reálnej aplikácií navrhujeme tieto úpravy:

* Zabezpečenie predspracovania dát do vyžadovaného formátu.
* Vynechanie kroku **A2.** V prípade reálnej aplikácie by bolo samotné generovanie odporúčaní realizované oddeleným komponentom. Navrhovaná metóda by teda zabezpečila len generovanie vzorov správania. Ich aplikácia by bola záležitosťou ďalších komponentov, ktoré by na požiadanie získali aktuálnu množinu vzorov (dátové objekty **O4** a **O5**). Samotné odporúčanie je v takom prípade oddelené od procesu hľadania vzorov. Ešte počas sedenia používateľa sa posielajú dáta o sedení (množina doteraz navštívených stránok) do časti softvéru, ktorá na požiadanie zabezpečuje odporúčanie pomocou aktuálnej množiny vzorov správania získanej druhou časťou softvéru, ktorá spracúva kompletné dáta o sedeniach používateľov prichádzajúce v prúde. Samozrejme aplikácie objavených vzorov správania môžu byť rôzne (pozri časť 2.2).
* V prípade reálnej aplikácie je možné prispôsobiť časy makrozhlukovania mikrozhlukov tak aby boli vykonávané v časoch nižšej prevádzky. To si ale vyžaduje identifikovať časy zníženej prevádzky a zmeniť konfiguráciu metódy.
* Naša metóda ráta so spracovávaním s oknom fixnej dĺžky. To je výhodné najmä čo sa týka jednoduchosti vyhodnocovania úspešnosti metódy na prúde dát simulovaného zo statického dataset. Je však pomerne jednoduché zmeniť metódu tak aby uvažovala okno s fixnou časovou dĺžkou.

### Návrh paralelizácie procesu

Navrhovaný proces pozostáva z viacerých komponentov, ktoré môžu fungovať nezávisle od seba. V princípe ide o rozdelenie prúdu dát na viacero vetiev. Každá transakcia z dátového prúdu je kopírovaná do jednotlivých vetiev, kde sa nezávislé od seba vykonávajú úlohy zhlukovania, hľadania globálnych vzorov a hľadania skupinových vzorov správania. Komponent pre zhlukovanie a komponent pre hľadanie skupinových vzorov zdieľajú medzi sebou modely používateľov s informáciou o ich priradení do zhlukov. Návrh je znázornený na obrázku 2.



Obrázok 2 Návrh paralelizácie procesu hľadania personalizovaných a globálnych vzorov

### Spôsob regulácie rýchlosti spracovania transakcií

Ako sme uviedli na začiatku kapitoly 4 venujúcej sa metódam dolovania vzorov správania z prúdových dát, tak je dôležité umožniť používateľovi nastaviť balans medzi rýchlosťou spracovania a presnosťou výsledkov. My sme sa rozhodli umožniť používateľovi nastaviť dolné obmedzenie požadovanej rýchlosti. Rýchlosť meriame ako počet spracovaných transakcií za jednu sekundu. Ak sa teda používateľ rozhodne, že je preňho spodná hranica rýchlosti spracovania napr. 20 transakcií za sekundu tak jednoducho nastaví toto obmedzenie a metóda sa bude týmto obmedzením riadiť. Čo sa týka výpočtovej náročnosti a vplyvu na toto obmedzenie je kritický krok aktualizácie uzavretých frekventovaných množín v algoritme dolovania frekventovaných množín *IncMine*. Táto aktualizácia nastáva vždy po každom novom prijatom segmente transakcií (viac o tomto algoritme v časti 4.1.2).

Naša metóda si v každom kroku udržiava informáciu o aktuálnej priemernej rýchlosti spracovania. 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:

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

Nech *tsupdate* označuje začiatok aktualizácie množín v *ms* (od začiatku *tstart*). Pred každou aktualizáciou sa vypočíta povolený maximálny čas na aktualizáciu v *ms* takto:

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

Vnútri samotnej aktualizácie množín:

1. Hľadajú sa uzavreté frekventované množiny v aktuálnom segmente pomocou algoritmu CHARM (opísaný v prílohe A). Použitie tohto algoritmu v rámci implementácie *IncMine* bolo rozhodnutím, ktoré vykonali autori implementácie algoritmu *IncMine* (Quadrana, Bifet, & Gavaldà, 2015)v rámci *MOA*, ktorú v mierných modifikáciach (opísaných v kapitole 6) používame. Rozhodli sme sa tomuto kroku prideliť časové obmedzenie vo veľkosti . Získaná množina korešponduje s množinou *F* opísanou v časti 4.1.2.
2. Druhá časť je krok, ktorý korešponduje s úlohou získania množiny *C* aktualizovaním množiny *L* semi-frekventovaných množín nad predchádzajúcim oknom pomocou množiny *F* získanej v prvom kroku (pozri časť 4.1.2). Na tento krok je vyhradený celý zostavajúci čas aktualizácie.

Tento podiel rozdelenia maximálneho času aktualizácie do jednotlivých krokov sme určili len na základe expertného odhadu a teda môže byť predmetom ďalšieho skúmania.

Jedná sa teda o celkom hrubý zásah do výpočtu, kedy niektoré vzory, ktoré by možno boli zaujímavé nebudú nájdené. Jedinou úpravou, ktorú sme urobili je zmena stratégie zoradzovania uzlov v prehľadávanom strome v algoritme CHARM zo vzostupej na zostupnú podľa podpory. Hoci autori algoritmu túto stratégiu označili ako menej efektívnu tak v tomto prípade môže v prípade predčasného ukončenia prehľadávania zabezpečiť že sa skôr nájdu vzory s vyššou podporou. Musí teda dôjsť k určitému kompromisu, kedy znižovanie požiadavky na minimálnu rýchlosť zvyšuje presnosť algoritmu. V kapitole 7 sa venujeme aj vyhodnoteniu tohto kompromisu a zisťujeme či takéto znižovanie presnosti má alebo nemá fatálne dôsledky konkrétne v aplikácií na odporúčanie.

Ďalším 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.

## Sumarizácia vstupných parametrov metódy

Keďže navrhovaná metóda je vlastne kombináciou viacerých metód dolovania v prúde dát, z ktorých každá má svoje vstupné parametre, a samotná metóda má vlastné parametre je celkový počet vstupných parametrov pomerne veľký. Počet parametrov je jedným z kritických častí navrhovanej metódy. V tejto časti v krátkosti opíšeme účel jednotlivých parametrov.

Vstupné parametre sme rozdelili do 4 kategórií podľa ich účelu:

* Parametre algoritmu hľadania uzavretých frekventovaných množín *IncMine*,
* Parametre algoritmu zhlukovania *CluStream*,
* Parametre odporúčania,
* Ostatné všeobecné parametre a obmedzenia metódy.

**Parametre algoritmu** **hľadania frekventovaných množín (IncMine):**

* **Minimálna podpora** *(skr. ms z ang. minimal support)*: Hodnota minimálnej podpory. V zmysle výpočtu progresívnej funkcie minimálnej podpory ako sme uviedli v časti 4.1.2 je to vlastne hodnota .
* **Miera uvoľnenia** *(skr. rr z ang. relaxation rate)*:Ako sme uviedli v časti 4.1.2 je to vlastne hodnota parametra *r*, teda určuje rýchlosť uvoľnovania potenciálne frekventovaných množín*.*
* **Dĺžka segmentu** *(skr. sl z ang. segment length)*: Je to počet transakcií, ktoré patria do jedného segmentu. Segment je ekvivalentom časovej jednotky tak ako sme to uviedli v časti 4.1.2. Kvôli zjednodušeniu testovania používame okno pevne fixovanej dĺžky namiesto okna s fixovaným časovým intervalom.
* **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 (Clustream):**

* **Počet zhlukov** *(skr. gc z ang. groups count)*: 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 je inkrementované s každým ďalším sedením používateľa. Ak toto počítadlo zmien dosiahne hodnotu vyššiu ako je hodnota tohto parametra tak sa počítadlo vynuluje a zmeny sa prenesú do štatistickej reprezentácie zhlukov v podobe mikrozhlukov. Tento parameter zároveň určuje kapacitu radu posledných sedení v modeli používateľa  *u* (atribút *u.niq,*  pozri časť 5.1.3).
* **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 sa tieto zmeny prenesú aj do do štatistickej reprezentácie zhlukov – tzv. mikrozhlukov. Počítadlo zmien v mikrozhlukoch je inkrementované s každým vykonaním tohto kroku o 1. Ak prekročí počet zmien v mikrozlukoch hodnotu danú týmto parametrom tak sa vynuluje a vykoná sa makrozhlukovanie.
* **Maximálny počet mikrozhlukov** *(skr. mmc z ang. maximal microclusters count)*: Viac o mikrozhlukoch v časti 4.2.1.

**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 na odporúčanie.
* **Počet odporúčaných položiek** *(skr. rc z ang. recommendations count)*.

**Ostatné všeobecné parametre metódy:**

* **Minimálna rýchlosť** *(skr. mts z ang. minimal transactions per second)*: Minimálna požiadavka na rýchlosť (meraná v jednotkách 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 identifikátorom makrozhlukovania *u.lmid* v modeli používateľa *u* a identifikátorom aktuálneho makrozhlukovania *lmid*. Ak teda platí *lmid – u.lmid > tcdiff* tak je model používateľa označený ako starý a odstránený.

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.

# Implementácia metódy

V tejto kapitole sa venujeme detailom implementácie metódy pre experimentálne účely. V úvode kapitoly rozoberáme aké externé moduly sme použili na implementáciu a ako sme ich integrovali do podoby navrhnutej v predchádzajúcej kapitole 5. Opisujeme tiež niektoré modifikácie týchto modulov, ktoré sme vykonali a opisujeme spôsob implementácie ďalších opisovaných funkcionalít.

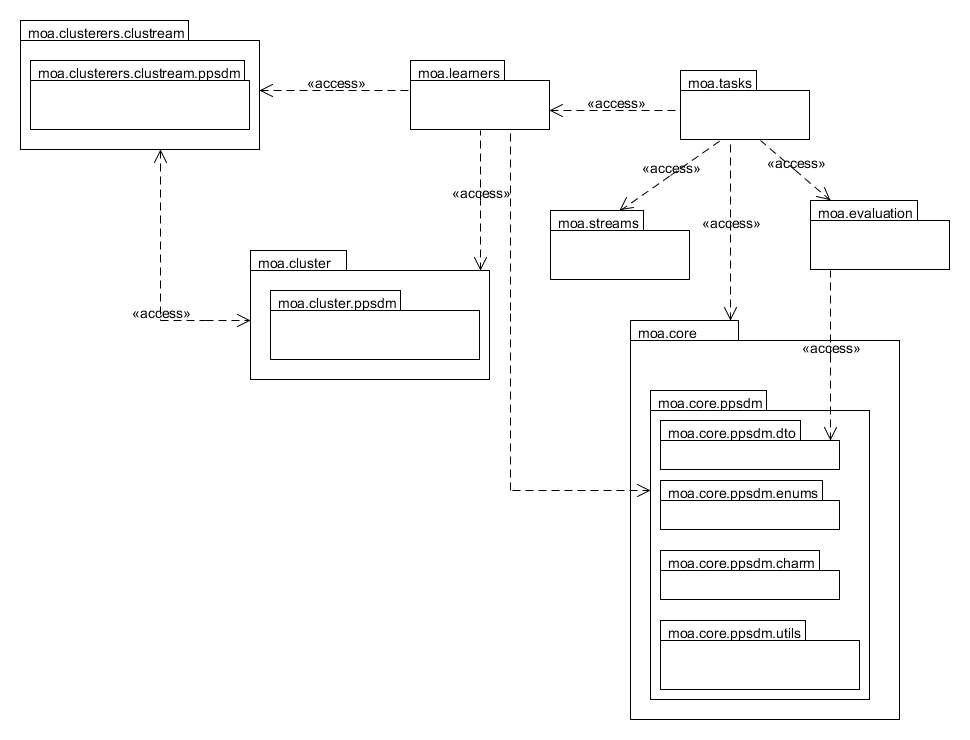
## Architektúra riešenia

Na implementáciu metódy sme sa rozhodli použiť rámec *MOA* (opísaný v časti 4.3.2) aj vďaka dostupnosti viacerých algoritmov dolovania v prúdových dátach. V tomto rámci je ako rozšírenie možné použiť implementáciu algoritmu *IncMine* tak ako ju opisujú v práci (Quadrana, Bifet, & Gavaldà, 2015). Autori práce implementujú tento algoritmus s niekoľkými rozdielmi oproti pôvodnej práci (Cheng, Ke, & Ng, 2008):

1. Algoritmus implementovali tak, že používa pohyblivé okno fixovanej dĺžky namiesto pohyblivého okna s fixovaným časovým intervalom tak ako v pôvodnej práci.
2. Na dolovanie uzavretých frekventovaných množín, čo je kritický bod čo sa týka celkovej výkonnosti algoritmu použili algoritmus CHARM, konkrétne implementáciu z rámca SFPM (Fournier-Viger, 2016). Je to modifikovaná verzia algoritmu, ktorá používa bitové množiny na reprezentáciu množín identifikátorov transakcií vo vertikálnom formáte dát.

Podobne je v tomto rámci dostupná aj implementácia algoritmu zhlukovania *Clustream* z makrozhlukovaním *k-means* (pozri časť 4.2.1), ktorú používame na zhlukovanie modelov používateľov. Tento algoritmus je súčasťou rámca MOA a objavil sa aj v práci (Hassani & Seidl, 2016).

Na Obrázok 3 možno vidieť diagram balíkov. Balíky, ktoré obsahujú v názve *ppsdm* sú tie, ktoré obsahujú triedy, ktoré sme vytvorili alebo modifikovali tak aby zabezpečovali funkčnosť navrhovanej metódy. Funkcionalitu sme integrovali do balíkov rámca *MOA*. V ďalších častiach sa budeme venovať jednotlivým balíkom, ich účelu a niektorým dôležitým triedam.



Obrázok 3 Diagram balíkov

### Balík moa.core

Tento balík je súčasťou rámca *MOA* kde obsahuje niekoľko veľmi dôležitých súčastí metódy PPSDM. Všetky tieto súčasti sme vložili do samostatného podbalíka s názvom *moa.core.ppsdm*.

#### Balík moa.core.ppsdm

V tomto balíku sa nachádzajú aj tieto triedy :

* *ConfigurationPPSDM* : statická konfigurácia metódy.
* *FciTablePPSDM* :

### Trieda PersonalizedIncMine

### Automatické čistenie tabuľky frekventovaných množín

Štruktúry na ukladanie a prácu so semi frekventovanými množinami v predchádzajúcej implentácií algoritmu IncMine neodstraňovali prázdne polia po uložených frekventovaných množínách v tabuľke ani polia s identifikátormi v invertovanom indexe. Hoci zabezpečili znovupoužitie týchto polí a nedochádzalo tak k neustálemu nárastu dátových štruktúr tak sme sa rozhodli vytvoriť mechanizmus, ktorý v pravidelných intervaloch tieto prázdne miesta odstráni. Tak ak aj dôjde k zmenám v prúde dát keď bolo napríklad nájdených mimoriadne veľa frekventovaných množín a následne sa počet nachádzaných frekventovaných množín znížil bude dátová štruktúra zmenšená a nebude obsahovať zbytočne veľa prázdnych miest.

### Oprava chyby s odstraňovaním položiek v zozname dobrých pozícií v tabuľke frekventovaných množín

Opravili sme chybu v implementácií algoritmu IncMine. Autori použili na zaznamenávanie dobrých pozícií (neprázdnych) v tabuľke frekventovaných množín ďalší zoznam. Položky v tomto zozname mazali nesprávne tak, že namiesto položky s uloženou pozíciou, ktorá sa mala vymazať vymazali položku na tejto pozícií v zozname. Dátovú štruktúru tohto zoznamu sme zmenili na množinu.

### Regulovanie rýchlosti spracovania

# Overenie metódy

Navrhovanú metódu overujeme z viacerých aspektov:

* úspešnosť aplikácie nájdených vzorov správania na úlohu odporúčania (7.1.1),
* vzťahy medzi skupinovými a globálnymi vzormi (7.1.2),
* rýchlosť spracovania prúdu dát (7.1.3).

Keďže navrhovaná metóda má pomerne veľké množstvo vstupných parametrov (pozri časť 5.3), ako vôbec prvý krok overenia vykonávame systematické hľadanie ich najlepšej konfigurácie, maximalizujúcej presnosť a rýchlosť metódy. Spôsob tohto hľadania opisujeme v časti 7.1.4. Ďalšie overenie potom vykonávame už na objavenej najlepšej konfigurácií.

Na overenie sme použili dva datasety z domén z odlišnými charakteristikami (e-learning a spravodajský portál), ktoré opisujeme v časti 7.1.5.

V časti 7.2 uvádzame vyhodnotenie priebehu hľadania najlepšej konfigurácie a získaných výsledkov. V časti 7.3 uvádzame podrobnejšie overenie vzťahov medzi globálnymi a skupinovými vzormi.

## Spôsob overenia jednotlivých aspektov a opis použitých dát

V tejto podkapitole opisujeme spôsob akým sú vyhodnocované jednotlivé aspekty riešenia, metodológiu navrhnutých experimentov a tiež charakteristiky použitých dát.

### Spôsob vyhodnotenia úspešnosti aplikácie na úlohu odporúčania

Stanovili sme počiatočnú hypotézu, ktorá je zameraná na aplikáciu vzorov správania v úlohe odporúčania. Jej znenie sme formulovali nasledovne:

*: 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.*

Odporúčanie budúcich návštívených stránok používateľovi sme vybrali ako jednu z možných aplikácií vzorov správania. Pomocou nej vyhodnocujeme či kombinovanie globálnych vzorov so skupinovými, čo je nosnou témou navrhovanej metódy, prináša zlepšenie a v akej miere. Úspešnosť odporúčania hodnotíme pomocou štandardných metrík *presnosť* (ang. precision), a *ndcg* (ang. normalized discounted cumulative gain). Konkrétnejšie sme tieto metriky zisťovali pre počty odporúčaných položiek 1, 2, 3, 4, 5, 10, 15, keďže v zvolených doménach je väčšina sedení pomerne krátka (pozri časť 7.1.5 kde sú opísané použité dáta).

Každá prichádzajúca transakcia T (pod transakciou rozumieme množinu akcií používateľa v rámci jedného sedenia – pozri časť 5.1.1) je rozdelená na trénovaciu množinu A (vyhodnocovacie okno) a testovaciu množinu B. Ak je dĺžka transakcie T menšia ako súčet veľkosti vyhodnocovacieho okna (parameter *ews*) a počtu odporúčaných položiek (parameter *rc*), tak sa pre vyhodnocovanie ignoruje. Nech počet odporúčaných položiek je N. Nech množina odporúčaných položiek je R. Nech H je bitový vektor dĺžky N, ktorý obsahuje 1 na i-tej pozícií iba ak i-ta odporúčaná položka z R patrí do prieniku .

Presnosť vypočítame ako:

|  |  |  |
| --- | --- | --- |
|  |  | (3) |

Metrika NDCG, nás bude zaujímať najmä pri väčšom počte odporúčaných položiek, kedy budeme sledovať aj to či správne odporúčané položky boli vybrané medzi prvými.

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

Kde IDCG je :

|  |  |  |
| --- | --- | --- |
|  |  | (5) |

DCG je :

|  |  |  |
| --- | --- | --- |
|  |  | (6) |

Aby bolo možné overiťstanovenú hypotézu tak sledujeme hodnoty metrík samostatne pre odporúčanie nami navrhovanou metódou využívajúcou kombináciu globálnych a skupinových vzorov (teda tak ako je opísaná v časti 5.1.4) (ozn. *GG*) a zároveň pre odporúčanie metódou využívajúcou len globálne vzory (ozn. *GL*) a len skupinové vzory (ozn. *GR*). Rozdiely medzi výsledkami týchto metód s rôznymi konfiguráciami vstupných parametrov sme nakoniec vyhodnotili a potvrdili ich signifikantnosť jednoduchým štatistickým t-testom.

### Spôsob vyhodnotenia vzťahov medzi skupinovými a globálnymi vzormi

Stanovili sme počiatočnú hypotézu, ktorá je zameraná na odhalenie prínosu skupinových vzorov správania v získavaní znalostí o správaní používateľov. Jej znenie sme formulovali nasledovne:

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

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

* počty unikátnych vzorov, ktoré sme vedeli získať pomocou našej 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 .

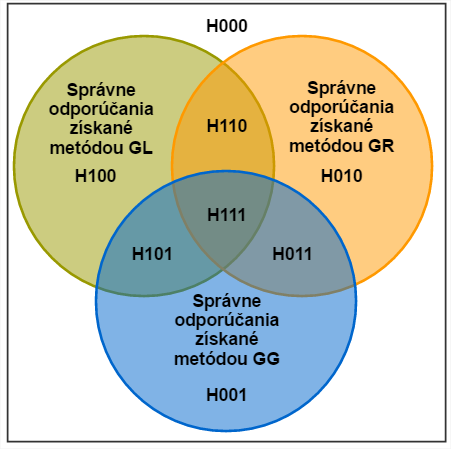
Okrem toho sledujeme počas behu experimentu pre každú transakciu počet správne odporúčaných položiek pre jednotlivé metódy odporúčania podľa toho aké vzory používajú. Jedná sa o tri samostatné množiny tak ako sú znázornené na vennovom diagrame (Graf 1). Ide o množiny všetkých správne odporúčaných položiek 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).

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 odporúčana položka bola nesprávna):

* Vektor úspešnosti odporúčania získaný pomocou metódy *GL*: **10001**.
* Vektor úspešnosti odporúčania získaný pomocou metódy *GR*: **11100**.
* Vektor úspešnosti odporúčania získany pomocou metódy *GG*: **01001**.

Rôzne prieniky týchto vektorov označujeme ďalej ako písmeno *H* (od 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*. Teda pre uvedený príklad by to bol prienik vektorov 10001 a 11100 čo je 10000. Hodnota *H101* je teda kardinalita výsledného vektora a to je 1. Pre jednoduchšie pochopenie sme znázornili uvedené množiny a ich prieniky pomocou vennovho diagramu (Graf 1).



Graf 1 Vennov diagram znázorňuje označenia jednotlivých množín vektorov úspešnosti odporúčania a ich prienikov.

Pomocou sumarizácie množín vektorov úspešnosti odporúčaní a ich prienikov pre všetky sedenia používateľov vieme zistiť nasledujúce fakty, ktoré nás zaujímajú:

* 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* aj *GG*: *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*
* Počet správnych odporúčaní pomocou *GG*: *H101+H011+H001+H111*
* Počet správnych odporúčaní pomocou *GL*: *H111+H110+H101+H100*
* Počet správnych odporúčaní pomocou *GR*: *H111+H011+H110+H010*

### Spôsob vyhodnotenia rýchlosti spracovania transakcií

Pod rýchlosťou spracovania transakcií myslíme počet transakcií spracovaných metódou za časovú jednotku. Metóda *GG* zahŕňa zhlukovanie v prude dát a hľadanie skupinových a globálnych vzorov správania zároveň. To prináša zvýšenú výpočtovú náročnosť oproti riešeniu, ktoré hľadá len globálne vzory správania (GL). V tejto časti vyhodnotenia sa zameriavame práve na tento aspekt.

Ako sme už pri návrhu metódy uviedli tak pokiaľ by išlo o reálnu aplikáciu bol by oddelený samostatný komponent pre hľadanie vzorov a samostatný pre generovanie odporúčaní alebo inú aplikáciu. Preto testujeme rýchlosť metódy samostne len pre úlohu hľadania vzorov.

Porovnávame teda priemernú rýchlosť metód *GG* a *GL* na najlepšej vybranej konfigurácií parametrov získanú z 20 nezávislých behov metódy. Z týchto metód sme odstránili komponent zabezpečujúci odporúčanie.

### Spôsob hľadania najlepších konfigurácií parametrov

Navrhovaná metóda kombinuje dve existujúce metódy dolovania znalostí z prúdu dát, ktoré používajú viacero parametrov, a sama vyžaduje od používateľa definovať hodnoty niekoľkých parametrov (pozri časť 5.3). 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 pri dodržaní minimálnej požiadavky na rýchlosť spracovania, ktorú sme stanovili na 15 transakcií za sekundu.

Aby sme zmenšili počet rôznych skúmaných konfigurácií tak sme parametre rozdelili do 4 kategórií podľa ich vzájomného súvisu tak ako sme uviedli v časti 5.3. V rámci každej kategórie parametrov skúmame vplyv zmien konkrétnych hodnôt parametrov na výsledky odporúčania. Z každej kategórie parametrov vyberieme konfigurácie, ktoré sú najsľubnejšie čo sa týka presnosti a zároveň rýchlosti spracovania.

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 parametrov. Zo získaných výsledkov vyberieme celkovo najlepšiu konfiguráciu.

Počiatočné nastavenie parametrov a voľbu testovaných hodnôt parametrov v jednotlivých skupinách sme získali iniciálnym odhadom experta. V Tabuľke 2 uvádzame rozdelenie parametrov do skupín a testované nastavenia parametrov.

Nech *h* je maximálny počet aktuálnych sedení, ktoré sú metódou udržované v pamäti. Vypočíta sa ako: . Vyhodnocovanie každej konfigurácie začína až po počte spracovaných sedení, vypočítanom ako: . Kde C je množina všetkých prehľadávaných konfigurácií parametrov. Eliminujeme tak rozdiely jednotlivých konfigurácií na začiatku spracovania prúdu, kedy konfigurácie s menšou dĺžkou *h* dokážu odporúčať skôr ako konfigurácie s väčšou dĺžkou *h*.

Dôležitým obmedzením v našom testovaní je obmedzenie rýchlosti (pozri časť 5.2.3). Rozhodli sme sa pre každú testovanú konfiguráciu fixovať minimálnu povolenú hodnotu rýchlosti (parameter *mts*) na 15 transakcií/sekunda.

Tabuľka 2: Testované nastavenia parametrov pri hľadaní najlepšej konfigurácie.

|  |  |  |
| --- | --- | --- |
| **parameter** | **kategória** | **testované hodnoty** |
| *ms* | *IncMine (dolovanie vzorov)* | 0.005, 0.01, 0.02, 0.03 0.04, 0.05, 0.1 |
| *rr* | *IncMine (dolovanie vzorov)* | 0.1, 0.5, 0.9 |
| *sl* | *IncMine (dolovanie vzorov)* | 25, 50, 100, 150, 200, 500 |
| *ws* | *IncMine (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 |

### Použité dáta

Na vyhodnotenie sme použili 2 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.

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. Použité dáta sme ďalej transformovali do požadovaného dátového formátu (pozri časť 5.1.1). V Tabuľka 3 uvádzame základné charakteristiky týchto datasetov.

Tabuľka 3: Základné charakteristiky datasetov.

|  |  |  |
| --- | --- | --- |
|  | **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). Nad týmito kategóriami rozoznávame vzory správania používateľov. Táto abstrakcia prináša tiež (ako je možné vidieť neskôr v časti 7.3) 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 Graf 2 a Graf 3 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, stredne veľa položiek s pomerne veľkou návštevnosťou a množstvo položiek navštevovaných len sporadicky.

Graf 2: Frekvencia návštev stránok používateĺmi v datasete ALEF. Na osi x sú stránky zoradené zľava doprava od stránky s najvyššou návštevnosťou po stránku s najmenšou návštevnosťou.

Graf 3: Frekvencia návštev kategórií stránok používateĺmi v datasete SP. Na osi x sú stránky zoradené zľava doprava od stránky s najvyššou návštevnosťou po stránku s najmenšou návštevnosťou.

Na Graf 4 a Graf 5 je znázornená distribúcia počtu sedení používateľov. Z hľadiska našej metódy je vhodné aby sa používatelia vracali na portál čo najviac. V tom prípade je nachádzaných tiež viac zaujímavých vzorov správania vzorov 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 6 a Graf 7). 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.

Graf 4: Histogram počtov sedení používateľov v ALEFe.

Graf 5: Histogram počtov sedení používateľov v spravodajskom portáli.

Graf 6: Histogram dĺžok sedení v ALEFe. Predspracovaný dataset ALEF s odstránením krátkych sedení (1 položka).

Graf 7: Histogram dĺžok sedení používateľov v SP. Predspracovaný dataset SP s odstránením krátkych sedení (1-2 položky).

## Vyhodnotenie úspešnosti odporúčania pre jednotlivé kategórie parametrov

V tejto časti vyhodnocujeme úspešnosť odporúčania pre rôzne konfigurácie vstupných parametrov metódy. Samostatne vyhodnocujeme jednotlivé kategórie parametrov uvedené v časti 5.3.

Na záver vyberáme najlepšiu konfiguráciu s ktorou pokračujeme v ďalších experimentoch. V prílohe E uvádzame dodatočné materiály k vyhodnoteniu (grafy, tabuľky). Detailné výsledky uvádzame v prílohe na elektronickom médiu.

### Vyhodnotenie pre kategóriu parametrov algoritmu hľadania frekventovaných množín (*IncMine*)

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

Na základe metriky *presnosť* sme vybrali najlepšie konfigurácie uvedené v Tabuľka 4.

Tabuľka 4: Hodnoty skúmaných parametrov najlepších konfigurácii kategórie *dolovanie vzorov - IncMine*. Ich rýchlosť v transakciách za sekundu a *presnosť* pre rôzne počty odporúčaných položiek. Sú to konfigurácie, ktoré sa umiestnili na prvých dvoch miestach v rámci metriky *presnosť* pre rôzne počty odporúčaných položiek. *ms* je minimálna podpora, *rr* je miera uvoľnenia, *sl* je dĺžka segmentu, *mil* je maximálna dĺžka množiny, *ws* je veľkosť okna. Intenzita farby znázorňuje najlepšie výsledky v rámci daného stĺpca.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **ms** | **rr** | **sl** | **mil** | **ws** | **rýchlosť**  **(t/s)** | **P@1** | **P@2** | **P@3** | **P@4** | **P@5** | **P@10** | **P@15** |
| **ALEF** | | | | | | | | | | | | | |
| **1** | 0.05 | 0.1 | 25 | 10 | 15 | 61.5 | 48.0% | 48.2% | 48.0% | 47.9% | 47.5% | 46.5% | 44.3% |
| **2** | 0.04 | 0.5 | 50 | 10 | 10 | 15.4 | 47.7% | 48.8% | 48.5% | 48.3% | 48.2% | 47.3% | 45.8% |
| **3** | 0.03 | 0.5 | 50 | 10 | 10 | 15.5 | 47.9% | 48.3% | 48.0% | 47.8% | 47.7% | 46.8% | 45.3% |
| **4** | 0.05 | 0.5 | 25 | 10 | 15 | 15.4 | 47.9% | 48.7% | 48.4% | 48.3% | 48.1% | 47.1% | 45.7% |
| **5** | 0.04 | 0.1 | 50 | 10 | 10 | 15.5 | 47.8% | 48.3% | 48.5% | 48.2% | 48.2% | 47.1% | 45.6% |
| **SPRAVODAJSKÝ PORTÁL** | | | | | | | | | | | | | |
| **1** | 0.05 | 0.5 | 25 | 10 | 15 | 3892.3 | 64.3% | 53.2% | 55.2% | 55.1% | 54.2% | 49.9% | 41.8% |
| **2** | 0.03 | 0.1 | 50 | 10 | 15 | 3065.1 | 64.0% | 53.3% | 55.5% | 55.8% | 55.8% | 57.2% | 50.9% |
| **3** | 0.005 | 0.1 | 500 | 10 | 15 | 574.2 | 52.8% | 45.3% | 49.5% | 52.5% | 56.3% | 69.7% | 72.1% |
| **4** | 0.05 | 0.1 | 25 | 10 | 15 | 3623.3 | 64.3% | 53.2% | 55.3% | 55.2% | 54.5% | 51.4% | 43.9% |
| **5** | 0.03 | 0.1 | 50 | 10 | 10 | 3812.2 | 63.9% | 53.1% | 55.4% | 55.6% | 55.2% | 53.8% | 46.7% |
| **6** | 0.04 | 0.1 | 50 | 10 | 15 | 3409.1 | 64.0% | 53.1% | 55.3% | 55.6% | 55.4% | 55.6% | 48.2% |
| **7** | 0.01 | 0.1 | 150 | 10 | 15 | 1688.4 | 60.9% | 52.0% | 54.1% | 55.1% | 56.3% | 64.5% | 64.4% |
| **8** | 0.005 | 0.5 | 500 | 10 | 15 | 999.3 | 57.2% | 49.9% | 51.7% | 53.0% | 56.1% | 68.9% | 69.8% |

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 uvoľnovania (*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.

Čo sa týka rýchlosti spracovania transakcií je vidno značný rozdiel v rýchlosti medzi datasetmi. Už sme uviedli (v časti venujúcej sa opisu dát 7.1.5), že tento efekt je spôsobený rozdielom v počte položiek nad ktorými sa hľadajú vzory správania.

V datasete ALEF všetky vybrané konfigurácie okrem konfigurácie 1 spadli až na spodnú hranicu minimálnej požadovanej rýchlosti (*mts =* 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.

### Vyhodnotenie úspešnosti odporúčania pre kategóriu parametrov zhlukovania

V tejto časti sa venujeme vyhodnoteniu úspešnosti odporúčania s rôznymi konfiguráciami kategórie parametrov týkajúcich sa úlohy zhlukovania (120 rôznych konfigurácií).

Vybrali sme najlepšie konfigurácie uvedené v Tabuľka 5.

Tabuľka 5: Hodnoty skúmaných parametrov najlepších konfigurácii kategórie *zhlukovanie*. Ich rýchlosť v transakciách za sekundu a *presnosť* pre rôzne počty odporúčaných položiek. Sú to konfigurácie, ktoré sa umiestnili na prvých dvoch miestach v rámci metriky *presnosť* pre rôzne počty odporúčaných položiek. *gc* je počet skupín, *tuc* je hraničný počet zmien v modeli používateľa, *tcm* je hraničný počet zmien v mikrozhlukoch, *mmc* je maximálny počet mikrozhlukov. Intenzita farby znázorňuje najlepšie výsledky v rámci daného stĺpca.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **gc** | **tuc** | **tcm** | **mmc** | **rýchlosť (t/s)** | **P@1** | **P@2** | **P@3** | **P@4** | **P@5** | **P@10** | **P@15** |
| **ALEF** | | | | | | | | | | | | |
| **1** | 6 | 5 | 400 | 100 | 50.7 | 48.8% | 48.8% | 48.6% | 48.2% | 47.9% | 47.0% | 45.1% |
| **2** | 8 | 5 | 400 | 100 | 52.7 | 48.7% | 48.7% | 48.6% | 48.4% | 48.0% | 46.9% | 44.9% |
| **3** | 8 | 5 | 800 | 100 | 58.4 | 48.4% | 48.7% | 48.4% | 48.0% | 47.8% | 46.7% | 44.8% |
| **4** | 4 | 5 | 400 | 100 | 47.5 | 48.6% | 48.5% | 48.3% | 48.0% | 47.8% | 46.6% | 44.6% |
| **5** | 4 | 5 | 200 | 100 | 54.0 | 48.6% | 48.6% | 48.3% | 48.2% | 47.8% | 46.8% | 44.8% |
| **6** | 8 | 5 | 200 | 100 | 54.0 | 48.5% | 48.5% | 48.5% | 48.2% | 48.0% | 46.9% | 44.9% |
| **7** | 6 | 5 | 200 | 100 | 47.7 | 48.3% | 48.4% | 48.4% | 48.1% | 47.9% | 46.9% | 44.9% |
| **8** | 4 | 5 | 200 | 1000 | 56.9 | 48.2% | 48.5% | 48.3% | 48.1% | 47.9% | 46.7% | 44.9% |
| **SPRAVODAJSKÝ PORTÁL** | | | | | | | | | | | | |
| **1** | 8 | 5 | 800 | 1000 | 1963.1 | 64.5% | 53.8% | 55.9% | 56.4% | 56.3% | 58.0% | 51.2% |
| **2** | 8 | 5 | 400 | 1000 | 1994.8 | 64.5% | 53.7% | 55.9% | 56.3% | 56.3% | 57.7% | 51.3% |
| **3** | 6 | 5 | 400 | 100 | 2869.3 | 64.3% | 53.4% | 55.5% | 55.9% | 56.0% | 57.7% | 51.4% |
| **4** | 6 | 5 | 800 | 1000 | 2018.3 | 64.5% | 53.7% | 55.8% | 56.2% | 56.2% | 57.9% | 51.2% |

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 veľky aby sa zachytilo potrebné množstvo informácií na vytvorenie správnych zhlukov používateľov.

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 konfiguráciach z kategórie parametrov *dolovanie vzorov - IncMine.*

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.

### Vyhodnotenie úspešnosti odporúčania pre kategóriu 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, ale sledujeme implicitne pri každej konfigurácií aj v ostatných kategóriach parametrov.

Nech je dĺžka sedenia *S.* Pre vyhodnocovanie odporúčania sú použité len sedenia s dĺžkou . Preto zmeny hodnôt parametrov *ews* a *rc* menia tiež počet sedení, ktoré môžu byť použité na vyhodnocovanie. Nemôžme teda priamo porovnávať konfigurácie s rozdielnými hodnotami *ews* a *rc* (bolo by to možné iba ak by sme ignorovali veľmi veľký počet krátkych sedení).

Sledujeme, že v datasete ALEF sú najlepšie výsledky dosahované pre *ews >= 2* a *ews <= 6* (Graf 8)*.* Pre dataset SP s veľkým množstvom krátkych sedení, konfigurácie s *ews=1* prinášajú výnimočne dobré výsledky pri odporúčaní 1 položky. Pri odporúčaní viacerých položiek dosahujú lepšie výsledky s vyšším *ews* (Graf 9)*.*

Malé 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 (časť 7.2.5) 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.

Graf 8: Výsledky metriky presnosť pre rôzne veľkosti vyhodnocovacieho okna ews pre rôzne počty odporúčaných položiek v datasete ALEF.

Graf 9: Výsledky metriky presnosť pre rôzne veľkosti vyhodnocovacieho okna ews pre rôzne počty odporúčaných položiek v datasete SP.

### Vyhodnotenie úspešnosti odporúčania pre ostatné parametre metódy

V tejto časti sa venujeme vyhodnoteniu odporúčania v spojitosti s rôznymi konfiguráciami kategórie ostatných parametrov metódy (*všeobecné*).

V tejto kategórií sledujeme rôzne hodnoty parametra *tcdiff*. Rôzne hodnoty parametra pre obmedzenie minimálnej rýchlosti (*mts*) testujeme až v časti venujúcej sa vyhodnoteniu rýchlosti spracovania (7.3.2).

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 4, Graf 5). Celkovo k makrozhlukovaniu dochádza v ALEFe 9 krát a v SP 18 krát. Preto testujeme rôzne hodnoty *tcdiff* po túto hodnotu.

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.

Graf 10: Mierne rastúci charakter presnosti so zmenou hodnoty parametra *tcdiff* v datasete ALEF.

Graf 11: Mierne rastúci charakter presnosti so zmenou hodnoty parametra *tcdiff* v datasete SP.

### Celkové vyhodnotenie úspešnosti odporúčania pre najlepšie 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 Tabuľka 5. 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 dosahujé 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: Hodnoty skúmaných parametrov celkovo najlepších konfigurácii. Ich rýchlosť v transakciách za sekundu a *presnosť* pre rôzne počty odporúčaných položiek. Sú to konfigurácie, ktoré sa umiestnili na prvých miestach v rámci metriky *presnosť* pre rôzne počty odporúčaných položiek a rôzne hodnoty *ews*. Intenzita farby znázorňuje najlepšie výsledky v rámci daného stĺpca.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **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** |
| **ALEF** | | | | | | | | | | | | | | | | | | |
| **1** | 0.05 | 0.1 | 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 | 0.1 | 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 | 0.1 | 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 | 0.1 | 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 | 0.1 | 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 | 0.1 | 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 | 0.1 | 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 | 0.5 | 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 | 0.5 | 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 | 0.1 | 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 | 0.1 | 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** | 0.005 | 0.1 | 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** | 0.005 | 0.1 | 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 | 0.5 | 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 | 0.1 | 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** | 0.005 | 0.5 | 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** | 0.005 | 0.5 | 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 | 0.1 | 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** | 0.005 | 0.5 | 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** | 0.005 | 0.1 | 500 | 10 | 15 | 6 | 5 | 800 | 1000 | 15 | 605.8 | 32.69% | 42.28% | 51.06% | 57.80% | 63.19% | 73.28% | 74.74% |

Vyhodnotili sme tiež priemerné rozdiely medzi presnosťou metódy využívajúcej len globálne vzory (*GL*),  metódy využívajúcej len skupinové vzory (*GR*) a metódy využívajúcej kombináciu skupinových a globálnych vzorov (*GG*) (Tabuľka 6). Vykonali sme jednoduchý t-test medzi populáciami hodnôt výsledkov metriky *presnosť* všetkých vyhodnocovaných konfigurácií metód.

Pre dataset ALEF metóda *GG* dosiahla signifikantne vyššiu presnosť oproti *GL* (p < 0.0001, t=9.7153, df=7600 a rozdiel stredných hodnôt je 0.0165) a tiež oproti *GR* (p < 0.0001, t=103.2904, df=7600 a rozdiel stredných hodnôt je 0.2143).

Pre dataset SP metóda *GG* dosiahla signifikantne vyššiu presnosť oprotri *GL* (p < 0.0001, t=3.8976, df=12598 a rozdiel stredných hodnôt je 0.0086) a tiež oproti *GR* (p < 0.0001, t=239.6845, df=12598 a rozdiel stredných hodnôt je 0.4137).

Tieto výsledky potvrdené jednoduchým štatistickým t-testom súhlasia s hypotézou , ktorú sme uviedli na začiatku kapitoly.

Tabuľka 7: Presnosti najlepších konfigurácií parametrov pre jednotlivé metódy. *GG* označuje metódu používajúcu globálne aj skupinové vzory správania, *GL* metódu používajúcu len globálne vzory a *GR* metódu používajúcu len skupinové vzory.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **P@1** | **P@2** | **P@3** | **P@4** | **P@5** | **P@10** | **P@15** |
| **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%**  **37.28%** |
| ***GG - GR*** | **51.82%** | **42.82%** | **40.25%** | **35.78%** | **31.84%** | **39.34%** | **37.28%** |

## Výsledné vyhodnotenie

V tejto časti pre najlepšie konfigurácie parametrov objavené v predchádzajúcej časti podrobnejšie vyhodnocujeme vzťahy medzi výsledkami (presnosť, rýchlosť) jednotlivých metód (*GL*, *GR*, *GG*).

### Vyhodnotenie vzťahov medzi globálnymi a skupinovými vzormi správania

Sledujeme rôzne prieniky vektorov úspešnosti odporúčania získaných pomocou metódy *GL*, *GR* a *GG*. V  Tabuľka 8 sú uvedené presnosti rôznych prienikov a zjednotení množín odporúčaní z týchto metód (pre vybranú najlepšiu konfiguráciu z vyhodnotenia úspešnosti odporúčania v predchádzajúcej časti 7.2.1). Pre pripomenutie označenie *H* jenasledované troma bitmi označujúce v poradí tri množiny správnych odporúčaní získaných metódou *GL*, *GR* a *GG*. Napr. označenie *H111* označuje spoločný prienik správnych odporúčaní zo všetkých troch metód.

V aplikácií našej metódy sa snažíme kombinovať skupinové a globálne vzory. V Tabuľka 8 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 skupinové vzory a globálne vzory nie (*H010+H011*). Toto je spôsobené tým, že v skupinách nevznikalo tak vela silných vzorov (s vysokou podporou) ako v globálnom priestore, keďže aj príslušné dátové vzorky boli menšie. Sledovali sme že presnosť odporúčania *GR* je vyššia v skupinách používateľov s väčším počtom aktívnych používateľov.

V pravidelných diskrétnych meraniach v čase spracovávania prúdu sme sledovali distribúciu počtu používateľov (Graf 12 a Graf 13), počty unikátnych vzorov v skupinách a presnosť jednotlivých metód. Merania boli vykonané pravidelne tesne pred vykonaním ďalšieho makrozhlukovania (je to 9 meraní v ALEFe a 18 meraní v SP). Počet používateľov v skupinách v ALEFe kolíše výraznejšie ako v SP 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.

Tabuľka 8 : Hodnoty presnosti pre rôzne počty odporúčaných položiek a rôzne prieniky metód GG, GR, GL.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Opis** | **Výpočet** | **ALEF** | | | | | | | **SPRAVODAJSKÝ PORTÁL** | | | | | | |
| [**P@1**](mailto:P@1) | **P@2** | **P@3** | **P@4** | **P@5** | [**P@10**](mailto:P@10) | [**P@15**](mailto:P@15) | **P@1** | **P@2** | **P@3** | **P@4** | **P@5** | [**P@10**](mailto:P@10) | [**P@15**](mailto: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%** |
| ***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%** |

Graf 12: Počet používateľov v jednotlivých skupinách v datasete ALEF v diskrétnych meraniach v čase počas behu spracovania dát.

Graf 13: Počet používateľov v jednotlivých skupinách v datasete SP v diskrétnych meraniach v čase počas behu spracovania dát.

Na Graf 14 a Graf 15 sledujeme aký je pomer počtu unikátnych vzorov v skupinách ku celkovému počtu vzorov. Vzor považujeme za unikátny ak sa taký vzor nevyskytuje v žiadnej inej skupine ani v množine globálnych vzorov. Tento pomer sa drží vždy nad hranicou 50%. Teda vždy aspoň polovica všetkých vzorov sú unikátne vzory. Každá skupina obsahuje v každej snímke minimálne 30% unikátnych vzorov naprieč skupinami. Tu sa nám už otvára odpoveď na stanovenú hypotézu na začiatku kapitoly, že skupinové vzory dokážu odhaliť nový pohľad na správanie používateľov v podobe unikátnych vzorov správania oproti globálnym vzorom správania.

Graf 14: Pomer počtu unikátnych vzorov v skupinách k celkovému počtu vzorov v diskrétnych meraniach v čase počas behu spracovania dát. Dataset ALEF.

Graf 15: Pomer počtu unikátnych vzorov v skupinách k celkovému počtu vzorov v diskrétnych meraniach v čase počas behu spracovania dát. Dataset Spravodajský portál.

Na Graf 16 a Graf 17 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.

Graf 16: Priemerná podpora všetkých vzorov a skupinových vzorov v diskrétnych meraniach v čase behu spracovania dát. Dataset ALEF.

Graf 17: Priemerná podpora všetkých vzorov a skupinových vzorov v diskrétnych meraniach v čase behu spracovania dát. Dataset Spravodajský portál.

Okrem celkovej presnosti metód sme pozorovali aj ich úspešnosť vo vnútri jednotlivých identifikovaných skupín používateľov. Uvádzame priemernú presnosť vo vnútri skupín (Graf 18 a Graf 19).

Graf 18: Priemerná presnosť odporúčania vo vnútri skupín používateľov v datasete SP. Presnosť je vypočítaná ako priemer presností vo vnútri jednotlivých skupín.

Graf 19: Priemerná presnosť odporúčania vo vnútri skupín používateľov v datasete ALEF. Presnosť je vypočítaná ako priemer presností vo vnútri jednotlivých skupín.

Z týchto výsledkov možno vidieť že metóda *GG* dosahuje v oboch datasetoch najvyššiu presnosť. V datasete ALEF metóda *GL* dosahuje lepšiu presnosť ako *GR*, zatiaľčo 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.

### Vyhodnotenie rýchlosti spracovania transakcií

Okrem presnosti odporúčania hodnotíme tiež rýchlosť spracovania prúdu dát. Aktuálnu rýchlosť (priemerný počet spracovaných transakcií za sekundu) logujeme v pravidelných časových intervaloch. Na Graf 20 a Graf 21 je možno vidieť porovnanie priemerných rýchlostí metód *GL* a *GG* (so zhlukovaním a hľadaním skupinových vzorov) z 20 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 udržateľný. Paralelizovanie navrhnutého procesu ako sme navrhli v časti 5.2.2, by malo viesť k zmierneniu tohto rozdielu. Môžeme tiež vidieť, že výpočet

Graf 20: Rýchlosť spracovania prúdu dát v ALEFe. Porovnanie metód GL a GG.

Graf 21: Rýchlosť spracovania prúdu dát v Spravodajskom portáli. Porovnanie metód GL a GG.

Okrem toho sme testovali účinok zvyšovania podmienky minimálnej požadovanej rýchlosti (vstupný parameter *mts*) na zmenu *presnosti*. Na grafe Graf 22 je znázornené k akému znižovaniu presnosti 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 (Graf 21). Maximálna sledovaná rýchlosť pri ktorej je metóda *GG* úspešnejšia ako *GL* je 1910 transakcií za sekundu. Úspešnosť metódy *GG* klesá v oboch datasetoch rovnako rýchlo ako *GR.*

V závislosti od požiadaviek aplikácie je teda možné prispôsobiť rýchlosť spracovania a požadovanej presnosti.

Graf 22: Graf znázorňuje vzťah medzi zvyšujúcou sa rýchlosťou spracovania (na horizontálnej osi) a znižujúcou sa presnosťou jednotlivých metód.

Graf 23: Graf znázorňuje vzťah medzi zvyšujúcou sa rýchlosťou spracovania (na horizontálnej osi) a znižujúcou sa presnosťou jednotlivých metód. Merania metód *GG* a *GR* sú zastavené pri maximálnej dosiahnutej rýchlosti.

# Záver a zhodnotenie

# Zoznam použitej literatúry

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*, 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). ACM.

Aggarwal, C. C., Bhuiyan, M. A., & Hasan, M. A. (2014). Frequent Pattern Mining Algorithms:A Survey. In V. E. Lee, R. Jin, & G. Agrawal, *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. (2004). A Framework for Projected Clustering of High Dimensional Data Streams. *Proceedings of the Thirtieth International Conference on Very Large Data Bases - Volume 30* (s. 852-863). Toronto, Canada: VLDB Endowment.

Aggarwal, C. C., Watson, T. J., Ctr, R., 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*, 81-92.

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

Agrawal, R., & Srikant, R. (1995). Mining sequential patterns. *Int'l. Conf. Data Engineering (ICDE 95)*, 3–14.

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*, 1472-1479.

Ayres, J., Flannick, J., Gehrke, J., & Yiu, T. (2002). Sequential PAttern Mining Using a Bitmap Representation. *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (s. 429-435). Edmonton, Alberta, Canada: ACM.

Barbará, D. (2003). Requirements for Clustering Data Streams. *ACM SIGKDD Explorations Newsletter*, 23-27.

Bayardo, J., & Roberto, J. (1998). Efficiently mining long patterns from databases. *ACM Sigmod Record*, 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. doi:10.1007/978-1-4939-0530-0\_10

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

Bockermann, C., & Blom, H. (2012). Processing Data Streams with the RapidMiner Streams Plugin. *Proceedings of the 3rd RapidMiner Community Meeting and Conference*.

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. (2014). Mining frequent itemsets in a stream. *Information Systems*, 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. *KDD '03 - Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 487 - 492.

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*, 191-215.

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

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

Etminani, K., Delui, A. R., Yenehsari, N. R., & Rouhani, M. (2009). Web Usage Mining: Discovery of the user's navagational patterns using SOM. *IEEE*, 244-249.

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*, 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., & Yin, Y. (2000). Mining Frequent Patterns Without Candidate Generation. *SIGMOD Rec.*, 1-12.

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). ACM.

Hassani, M., & Seidl, T. (2016). Using internal evaluation measures to validate the quality of diverse stream clustering algorithms. *Vietnam Journal of Computer Science*, 1--13.

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. doi:http://dx.doi.org/10.1016/j.ipm.2007.03.003

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. doi:http://dx.doi.org/10.1016/j.eswa.2010.02.105

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

Kranen, P., Assent, I., Baldauf, C., & Seidl, T. (2011). The ClusTree: indexing micro-clusters for anytime stream mining. *Knowledge and Information Systems*, 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*, 1-10.

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

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

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*, 16-21.

Mobasher, B., Dai, H., Luo, T., Sun, Y., & Zhu, J. (2000). Integrating Web Usage and Content Mining for More Efective Personalization. 165-176.

Morales, G. D., & Bifet, A. (2015). SAMOA: Scalable Advanced Massive Online Analysis. *Journal of Machine Learning Research*, 149-153.

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*, (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. *17th international conference on data engineering*, (s. 215-224).

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). 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*, 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 V. E. Lee, R. Jin, & G. Agrawal, *Frequent Pattern Mining* (s. 261-281).

Sisodia, D. S., & Verma, S. (2012). Web usage pattern analysis through web logs: A review. *JCSSE 2012 - 9th International Joint Conference on Computer Science and Software Engineering*, 49-53.

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

Srivastava, M., Garg, R., & Mishra, P. K. (2014). Preprocessing Techniques in Web Usage Mining: A Survey. *International Journal of Computer Applications (0975 – 8887), Volume 97*, 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*, 444-455.

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

Tzvetkov, P., Yan, X., & Han, J. (2005). TSP: Mining top-k closed sequential patterns. *Knowledge and Information Systems*, 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*, 372-382.

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

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

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

1. http://www.w3.org/TR/WD-logfile.html [↑](#footnote-ref-2)
2. napr. https://www.weblogexpert.com/info/IISLogs.htm [↑](#footnote-ref-3)
3. https://www.weblogexpert.com/ [↑](#footnote-ref-4)
4. https://www.webtrends.com [↑](#footnote-ref-5)