VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

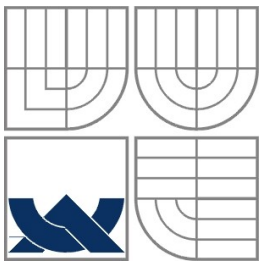
DOLOVÁNÍ ASOCIAČNÍCH PRAVIDEL

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

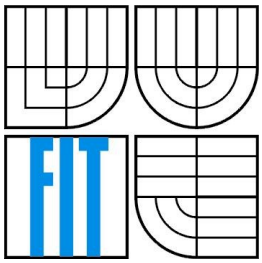
AUTOR PRÁCE
AUTHOR

ONDŘEJ ŠABATKA

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

DOLOVÁNÍ ASOCIAČNÍCH PRAVIDEL

ASSOCIATION RULE MINING

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR

ONDŘEJ ŠABATKA

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. VLADIMÍR BARTÍK, Ph.D.

BRNO 2008

Dolování asociačních pravidel

Zadání bakalářské práce

1. Seznamte se s problematikou získávání asociačních pravidel z dat.
2. Po dohodě s vedoucím práce zvolte algoritmus a navrhněte aplikaci, která by vhodným způsobem prezentovala funkčnost zvoleného algoritmu.
3. Navrženou aplikaci implementujte a ověřte její funkčnost na zvoleném vzorku dat.
4. Zhodnoťte dosažené výsledky a další možná pokračování tohoto projektu.

Licenční smlouva

Licenční smlouva je součástí originálu této práce a je uložena v archivu Fakulty informačních technologií Vysokého učení technického v Brně.

Abstrakt

Tato bakalářská práce se zabývá dolováním asociačních pravidel. První část se věnuje vysvětlení technologie dolování dat a teorie, kterou je dobré znát pro seznámení se s asociační analýzou. Další část se věnuje samotné asociační analýze a podrobně vysvětluje principy algoritmu Apriori. Poslední část práce popisuje implementaci a testování algoritmu Apriori v programovacím jazyce Java.

Klíčová slova

Dolování dat, dolování asociačních pravidel, asociační pravidla, asociační analýza, algoritmus Apriori

Abstract

This bachelor's thesis is concerned with the association rule mining. The first part is devoted to the explanation of data mining technology and theory, which are necessary pre-steps for getting acquainted with association analysis. The next part focuses on the association analysis itself and explains the principals of algorithm Apriori in detail. The last part of the thesis describes the implementation and testing of algorithm Apriori in the Java programming language.

Keywords

Data mining, association rule mining, association rules, association analyzis, Apriori algorithm

Citace

Šabatka Ondřej: Dolování asociačních pravidel. Brno, 2008, bakalářská práce, FIT VUT v Brně.

Dolování asociačních pravidel

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Vladimíra Bartíka, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

…….....……
Jméno Příjmení, datum

Poděkování

Děkuji Ing. Vladimíru Bartíkovi, Ph.D. za odborné vedení a za ochotu pomoci při řešení problémů, které nastaly při vypracovávání této práce. Dále chci také poděkovat za poskytnutí databáze z obchodního domu Globus pro potřeby testování programu. Také děkuji Danielu Řepovi za vysvětlení základů programovacího jazyka Java.

Obsah

Obsah.....1
Úvod.....3
1 Definice pojmů.....5
1.1 Data, informace, znalosti.....5
1.2 Získávání znalostí z databází.....5
1.3 Dolování dat.....6
1.4 Datový sklad.....7
2 Teoretický úvod.....8
2.1 Proces získávání znalostí.....8
2.2 Zdroje dat pro dolování.....9
2.3 Předzpracování dat.....10
2.4 Metody dolování dat.....10
2.4.1 Klasifikace a predikce.....11
2.4.2 Shluková analýza.....11
2.4.3 Neuronové sítě.....11
2.4.4 Rozhodovací stromy.....12
3 Asociační analýza.....13
3.1 Asociační pravidla.....13
3.2 Stanovení relevance pravidla.....14
3.3 Rozdělení asociačních pravidel.....15
4 Dolování asociačních pravidel.....16
4.1 Úvod do problematiky.....16
4.2 Triviální vyhledávání.....17
4.3 Algoritmus Apriori.....17
4.3.1 Dva kroky algoritmu.....18
4.3.2 Funkce algoritmu.....18
4.3.3 Zvýšení efektivity algoritmu Apriori.....20
4.3.4 Modifikace algoritmu Apriori.....21
4.4 Shrnutí.....21
5 Praktická část.....22
5.1 Uvedení.....22
5.2 Návrh aplikace.....22
5.2.1 Main.java.....22
5.2.2 Loader.java.....23

5.2.3 Apriori.java 23
5.2.4 Třídy pro uložení dat 24
5.2.5 Htmlgenerator.java 24
5.3 Práce s programem 24
5.3.1 Uživatelem zadávaná data 24
5.3.2 Formát vstupních dat 25
5.3.3 Prezentace výsledků algoritmu Apriori 25
5.4 Testování programu 26
5.4.1 Ověření funkčnosti 26
5.4.2 Testy doby výpočtu algoritmu 27
6 Závěr 29
Literatura 30
Seznam příloh 31

Úvod

Současná doba je charakterizována intenzivním nasazováním informačních technologií do nejrůznějších odvětví lidské činnosti a díky tomu také narůstá množství různorodých informací, které se ukládají do databází. Vzhledem k neustálému rozvoji technologií a vzrůstajícím kapacitám úložných médií máme možnost zaznamenávat informace o nákupech zákazníků v obchodech, o situacích na burze, či bankovních operacích klienta. Může se zdát, že tato data nemohou mít žádné smysluplné využití, opak je ale pravdou. Díky tradičním přístupům (dotazovací jazyky jako například SQL) můžeme vytvářet přehledové sestavy a výkazy, dále můžeme využít agregačních funkcí k získání souhrnů. Takto může například obchodník zjistit kolik zboží bylo prodáno za minulé období nebo kolik činí jeho obrat nebo celkový zisk. Uložená data mohou ovšem pomoci odhalit mnohem více informací, které jsou na první pohled skryté. Pomocí analytických nástrojů můžeme vyhledávat například vztahy mezi prodávanými výrobky nebo můžeme na základě zdravotních záznamů analyzovat, zda má kouření neblahý dopad na kuřákovo zdraví.

Složité analytické metody nelze aplikovat na transakčních databázích, které jsou určeny k ukládání operačních údajů. Vysoká výpočetní náročnost těchto metod by vedla k nepřípustnému zatížení databáze. Často jsou také k analýze třeba historická data, která by zbytečně zpomalovala práci s operační databází. Proto moderní databázové servery podporují vytváření datových skladů a analytické technologie OLAP a data mining (dolování dat). Datové sklady umožňují integrovat velké množství dat z více zdrojů a vytvářet tak vhodné prostředí pro analýzu. Technologie OLAP definuje uložení velkých objemů dat v databázi tak, aby uspořádání bylo srozumitelné pro uživatele, který se zabývá jejich analýzou. Proces dolování dat slouží k vyhledávání užitečných informací ukrytých v datech.

Technologií dolování dat se dnes zabývá velké množství komerčních firem a mnoho podniků pomocí této technologie získává cenné informace pro své marketingové plánování. Marketing ovšem není jediným odvětvím, ve kterém se dolování dat v současné době nasazuje. Technologie byla také použita v molekulární biologii, konkrétně v genetice[1]. Dalšími vědními obory, ve kterých dolování dat našlo své uplatnění, jsou medicína či hutnictví.

Přesto je nejčastějším použitím technologie dolování dat oblast marketingu, kde tato technologie může pomoci vytypovat nové trhy a pomáhá při nacházení atraktivnějších prodejních míst. Například maloobchodní společnost REI, prodávající vybavení pro outdoor aktivity, používá software pro dolování dat k rozboru objemných údajů o zákaznících, které sbírá prostřednictvím své webové stránky, poštovního kontaktu a 78 obchodních domů. Pokud společnost REI začne uvažovat o nových místech pro své prodejny, prozkoumá data o internetových objednávkách, aby našla místa s vysokou koncentrací zákazníků, kteří již u společnosti nakupují on-line. Tato firma využívá nástroje

1 Bližší informace: Zeman, D.: Aplikace procesu dolování dat v biologii – genetice. Brno, VUT FIT, 2005.

dolování dat také k přizpůsobení sortimentu svých obchodů preferencím místního trhu a k odkrývání vzorců, které popisují budoucí nákupy zákazníků.[2]

Technologiemi dolování dat se zabývá mnoho velkých firem. Za zmínku stojí například SAS, SPSS, IBM, Computer Associates a Fair Isaac společně s desítkami menších specializovaných společností, které se o tuto novou oblast také ucházejí.[3]

Dolování dat je dnes jedním z nástrojů Business Intelligence, což je sada postupů a technologií, které mají firmám pomáhat a účelně je podporovat v rozhodovacích procesech. Představuje komplex aplikací, které podporují analytické a plánovací činnosti podniků a organizací a jsou postaveny na specifických, tzv. OLAP technologiích a jejich modifikacích.

Existuje velké množství metod pro dolování dat. Tato práce se zabývá jednou z nich, a to konkrétně asociační analýzou. Typickým příkladem této metody je analýza nákupního košíku, která se zabývá hledáním kombinací zboží vyskytujícího se v nákupech zákazníků významně často společně. Výsledky této metody pak mohou pomoci managementu při rozmísťování zboží v obchodě tak, aby byl prodej efektivnější.

První kapitoly práce se věnují vyjasnění používané terminologie a uvádí definice pojmů, které jsou používány v této práci. Následuje teoretický úvod, ve kterém je čtenář obeznámen s principy, kterým je dobré porozumět, aby se mohl blíže seznámit s analýzou asociací a dolováním asociačních pravidel. V této části práce je také nabídnut přehled využívaných metod pro dolování znalostí. Další kapitola se již podrobně věnuje asociační analýze a v následující kapitole jsou uvedeny některé algoritmy sloužící k dolování asociačních pravidel. Především je zde popsán algoritmus Apriori a jeho modifikace. V poslední kapitole je popis implementace a testování programu, který demonstruje principy algoritmu Apriori.

2 Cowley, S.: Dolování dat. 2005. Článek dostupný na URL http://archiv.computerworld.cz/-cwarchiv.nsf/clanky/5C3E3B4604AE67AFC125711700539FEA?OpenDocument (2008)

3 Ze stejného zdroje jako 17.

1 Definice pojmů

Vzhledem k tomu, že se tato práce zabývá specializovanou problematikou týkající se databází, je důležité uvést zde definice obecných pojmů a termínů, které se v práci vyskytují a které by bez zasazení do správného kontextu mohly být vykládány chybně.

1.1 Data, informace, znalosti

Data jsou jakékoli vyjádření (reprezentace) skutečnosti, schopné přenosu, uchování, interpretace či zpracování. Z hlediska počítačového jsou to pouze hodnoty různých datových typů.

Informace jsou data, která mají sémantiku (význam).

Znalosti jsou informace po jejich zařazení do souvislostí.[4]

Obrázek 1: Informační úrovně

Pro dolování dat je tedy zapotřebí sbírat a ukládat velké množství dat (občas je termín sběr dat zaměňován za sběr informací). Prezentací uložených dat vznikají informace, což je vlastně subjektivní interpretace dat uživatelem. Proto je naším cílem získávání znalostí, tedy informací zasazených do kontextu, které jsou uživatelem interpretovány vždy stejně. Takže znalosti můžeme chápat jako shluky informací s daným významem, které jsou pro člověka nějakým způsobem užitečné.

1.2 Získávání znalostí z databází

Nejprve je vhodné vysvětlit souvislost mezi získáváním znalostí a dolováním dat. Získávání znalostí je rozsáhlá disciplína, která zahrnuje mnoho součástí. Jednou z nich je právě dolování dat. Mnohdy ovšem bývá dolování dat (anglicky data mining) nesprávně používáno pro označení celé disciplíny získávání znalostí. Je to především díky kratšímu a snadněji zapamatovatelnému názvu. Podrobněji vztah mezi získáváním znalostí a dolováním dat vysvětluje kapitola 2.1.

4 Hruška, T.: Informační systémy, pokročilé informační systémy. Brno 2006, studijní opora, FIT VUT v Brně.

Zajímavou definici získávání znalostí z databází formuloval v roce 1996 pan Fayyad se svými kolegy: *„Získávání znalostí z dat je netriviální proces zjišťování platných, neznámých, potenciálně užitečných a snadno pochopitelných závislostí v datech.“*.[5]

Fakt, že se jedná o netriviální proces, nám říká, že získání dané znalosti nelze provést pomocí běžných nástrojů, a je proto potřeba využít sofistikovaných, výpočetně náročných, analytických metod. Déle nám definice říká, že se musí jednat o platnou a neznámou závislost v datech, čímž je míněno, že nehledáme informace na první pohled viditelné, zřejmé či získatelné pomocí konvenčních metod. Další podmínkou je, že získaná informace musí být platná, tedy musí být ověřitelná. Nakonec nám definice říká, že získaná znalost musí být potenciálně užitečná. To znamená, že tato znalost musí být nějakým způsobem využitelná (například při procesu rozhodování).

1.3 Dolování dat

Sám pojem dolování dat může na člověka působit poněkud neformálně až hanlivě, ale ve skutečnosti se již vžil do obecného povědomí. Často bývá dolování dat definováno jako proces vyhledávání skrytých závislostí a odlišností. Zjednodušeně se jedná o definici pravdivou, ale ne zcela přesnou. Formálních definic dolování dat existuje veliké množství. Zde jsou uvedeny dvě, které se v některých ohledech vhodně doplňují.

> *Data mining (dolování z dat či vytěžování dat) je analytická metodologie získávání netriviálních skrytých a potenciálně užitečných informací z dat.*[6]

> *Data mining je proces, který používá různé analytické nástroje pro vyhledávání skrytých vzorů a vztahů v datech, které mohou být použity jako podklad pro rozhodování.*[7]

Z uvedených definic vyplývá, že dolování dat je inteligentním procesem vyhledávání vzorů a vztahů v datech. Je to jeden z kroků procesu získávání znalostí. Tento proces bychom mohli rozdělit na dvě části a to na přípravu dat a na získávání a následnou prezentaci znalostí.

Za matematický základ dolování dat bývá často považována statistika, ale to není zcela přesné. Bylo tomu tak především proto, protože statistika představovala osvědčený prostředek pro modelování a analyzování závislostí v datech. Dolování znalostí je ale mnohem komplexnější disciplínou, která zahrnuje mnoho principů a metod a statistika je právě jednou z nich. Dolování dat se liší od statistických metod a metod umělé inteligence především tím, že je zde kladen důraz na přípravu dat pro analýzu a na prezentaci výsledných znalostí.

S pomocí dolování dat můžeme provádět řadu složitých analýz. Nejčastěji se v praxi setkáváme s požadavky na tři typy analýz a to na klasifikaci, předpověď a na odhalení závislostí a rozdílů. Klasifikace slouží k vyhodnocení stavu nějakého předmětu našeho zájmu na základě informací a dat,

5 Kocan, M.: Dolujeme a dolujeme. Červen 2006. Článek dostupný na URL http://www.dbsvet.cz/view.php?cisloclanku=2006060101 (březen 2008)

6 Definice pojmu data mining podle http://cs.wikipedia.org/

7 Two Crows Corporation, Potomac USA: Introduction to Data Mining and Knowledge Discovery. 2005

které o něm máme. Předpověď je odhadováním dalšího vývoje na základě stávajících informací v kombinaci s možností nastavit okolní vlivy podle předpokladu. Odhalení závislostí a rozdílů řeší vyhledávání vzájemných vazeb a odlišností neodpovídajících ostatním zpracovávaným datům.

Cílem dolování dat je vždy konkrétní řešení obchodního problému nebo hledáním cesty k vylepšení stávajícího procesu. Cíl je vždy třeba přesně předem definovat, protože je nutné předem připravit data pro zadanou úlohu.

Aby mělo dolování dat opravdu smysl, musíme zajistit, aby byly nalezené informace vhodně prezentovány a to tak, aby mohly být co nejjednodušeji pochopeny uživatelem. Používají se proto textové výstupy (tabulky, sestavy, přehledy, atd.) nebo vizualizace (sloupcové grafy, plošné grafy, atd.).

1.4 Datový sklad

Nejpoužívanější definice pochází od Billa Inmona, který říká: *„Datový sklad je podnikově strukturovaný depozitář předmětově orientovaných, integrovaných, časově neměnných, historických dat použitých na získávání informací a podporu v rozhodování. V datovém skladu jsou uložena atomická a sumární data.“.*[8]

V běžných databázích je snaha o co nejmenší redundanci dat. Datové sklady jsou orientované na předmět (například na zákazníka nebo na určité zboží), tedy se snaží o separaci jednotlivých funkčních celků tak, aby struktura skladu byla co nejlépe čitelná pro uživatele. Zvyšují se tak ale nároky na úložný prostor. Dále jsou datové sklady integrované, což znamená, že seskupují více datových zdrojů (často více operačních databází podniku). Data jsou do skladu nahrávána v určitých časových intervalech a po velkých kvantech. Uložená data se již nemění. V datových skladech jsou tedy nahrané údaje za dlouhé časové období, obvykle několika roků.

Definice datového skladu je zde uvedena z důvodu, protože se jedná o častý případ úložiště dat v procesu získávání znalostí z databází a v této práci je takto několikrát zmiňován (nejedná se ovšem o jedinou variantu).

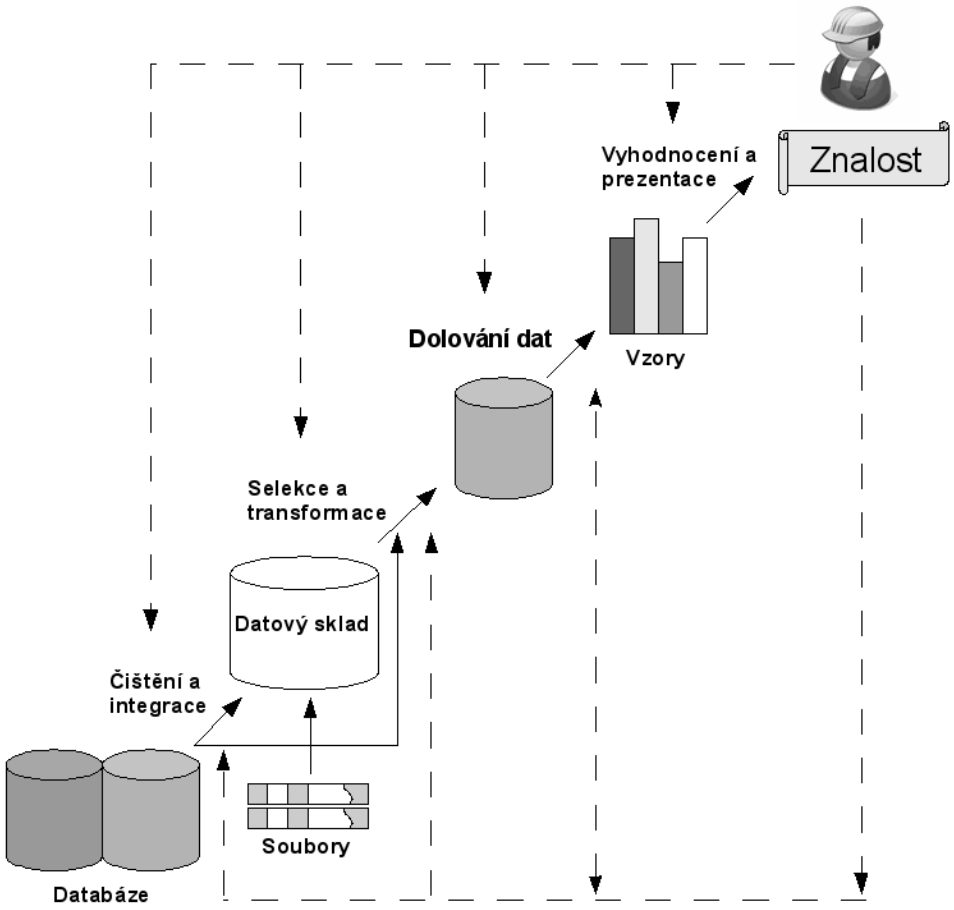
8 Lacko, L.: Databáze: datové sklady, analýza OLAP a dolování dat s příklady v Microsoft SQL Serveru a Oracle. Brno, Computer Press 2003, s. 48 - 49

2 Teoretický úvod

Tato kapitola čtenáře seznámí se skutečnostmi, které předcházejí samotnému dolování dat. První podkapitola nabízí pohled na získávání znalostí z databází jako na proces, který je sestaven z několika kroků, které se v jistých posloupnostech opakují. Další podkapitola naznačuje jaká data jsou vhodná pro dolování. Následně jsou v krátkosti naznačeny principy procesu předzpracování dat a v poslední podkapitole je k dispozici přehled častěji používaných metod dolování dat.

2.1 Proces získávání znalostí

Dříve již bylo zmíněno, že získávání znalostí z databází je velice rozsáhlá disciplína a dolování dat je jeho součástí. Na následujícím obrázku (*obrázek 2*) je znázorněn celý proces získávání znalostí a je zde ukázáno, jaké kroky se při něm provádějí a v jakém pořadí.



Obrázek 2: Proces získávání znalostí

1. **Čištění dat** – odstraňují se přebytečná a nekonzistentní data.
2. **Integrace dat** – je možné kombinovat více datových zdrojů. Často se kroky čištění a integrace dat provádějí společně a výsledek je ukládán do datového skladu.

3. **Selekce dat** – vybíráme data, která jsou pro danou analytickou úlohu relevantní.
4. **Transformace dat** – data se transformují do sjednocené podoby vhodné pro dolování pomocí sumarizačních a agregačních operací. Pokud jsou použity datové sklady, může předcházet tento proces selekci dat, protože bývá součástí tvorby datového skladu.
5. **Dolování dat** – inteligentní metoda používaná pro vyhledávání datových vzorů.
6. **Vyhodnocení vzorů** – podle nastavené míry užitečnosti se vyhledávají skutečně zajímavé vzory. Posoudit ovšem, zda je vzor zajímavým, není vůbec jednoduché. Existují 4 vlastnosti, kterými je zajímavost vzoru charakterizována:
    - srozumitelnost pro člověka
    - platnost pro nová a testovací data s určitým stupněm jistoty
    - potenciální užitečnost
    - novost

    Vzory, které můžeme označit za zajímavé, jsou znalostmi. Konkrétní způsob tohoto posuzování si ukážeme v kapitole popisující metodu asociační analýzy, kde se zajímavost pravidla určuje pomocí *podpory* a *spolehlivosti*.
7. **Prezentace znalostí** – vizualizace a prezentace nalezených znalostí uživateli.[9]

2.2 Zdroje dat pro dolování

V současné době se nejčastěji k dolování dat používají relační databáze. Data jsou v nich uspořádána do tabulek (relací), nad kterými jsou definovány operace. Uložení dat v tabulkách musí splňovat podmínku první normální formy, což znamená, že údaje v jednotlivých sloupcích tabulky musí být atomické. Tato vlastnost se ovšem pro dolování dat vždy nehodí. Proto pokud využíváme relační databáze, tak často také využíváme datové sklady, ve kterých se shromažďují data z operačních databází a můžeme na ně pohlížet z historické perspektivy.

Relační databáze nejsou ale jediným zdrojem dat. Těch může být mnoho, a proto následuje krátký přehled některých možných variant.

- Transakční databáze – v podstatě se jedná o soubor, kde každý řádek (záznam) reprezentuje jednu transakci.
- Textové databáze – pod tímto pojmem rozumíme kolekci dokumentů a pomocných struktur (datových souborů, např. XML). Dolováním například můžeme zjišťovat, jakým jazykem je dokument napsán.
- Multimediální databáze – obsahují multimediální data, tedy obrazová, video a audio data.
- Web – z určitého pohledu se jedná o velice rozsáhlou databázi s obrovským potenciálem pro dolování. Například můžeme automaticky třídit webové stránky podle jejich obsahu.
- a další

9 Han, J., Kamber, M., Data mining: Concepts and Techniques. USA, Academic Press 2001, s. 6 - 7

2.3 Předzpracování dat

Data, která se ukládají v operačních databázích podniků nejsou optimalizována pro analýzu. Jedná se spíše o podnikovou dokumentaci a archivaci údajů. Proto se u těchto dat můžeme setkat s problémy, které by případnou analýzu znemožnily. Například se může jednat o následující:[10]

- Data obsahují špatné údaje, způsobené chybami měřících přístrojů nebo chybou lidské obsluhy.
- Některé údaje jsou nevyplněné. U některých atributů se dokonce stává, že jsou vyplněny jen zcela výjimečně – v takových případech se hovoří o tzv. řídce obsazených atributech.
- Data jsou popsána příliš mnoha atributy a není zřejmé, které z nich jsou pro řešení dané úlohy relevantní. Úspěch závisí na volbě vhodné množiny atributů.
- Data mají formu složitého relačního schématu.

Předzpracování dat nám pomáhá tyto problémy řešit. Ve všech případech je žádoucí tyto nedostatky odstraňovat, což je zajišťováno pomocí několika kroků procesu získávání znalostí, které předcházejí samotnému dolování dat (viz. *obrázek 2*). Jedná se o čištění, integraci, selekci, redukci a transformaci dat.

Úkolem předzpracování dat je tedy odhalovat chyby v reálných datech, upozornit na ně a přijmout rozhodnutí, jak s nimi naložit. Například doplnit údaje ve spolupráci s expertem z oboru nebo při doplňování využít statistických zákonitostí. Dochází také k doplnění výchozích dat z dalších datových zdrojů. Také jsou redukovány nadbytečné dimenze (množství dat), protože s každou dimenzí se exponenciálně zvyšuje náročnost zpracování. Dále se data transformují do tvaru vhodného pro řešení dané dolovací úlohy.

2.4 Metody dolování dat

Pomocí techniky dolování dat můžeme řešit velké množství zdánlivě nesouvisejících úloh týkajících se mnoha oborů. Je to díky tomu, že dolování dat není jedinou ustálenou metodou, ale je to spíše označení skupiny metod. Dá se tedy říci, že dolování dat je vlastně metodologií, kde si analytik sám může vybrat nejvhodnější metodu pro získávání znalostí. Výběr této metody je jedním z prvních kroků v celém procesu a je třeba ho důkladně zvážit. Pro řešení jedné úlohy totiž může existovat metod více a je pak třeba tyto metody srovnat a vybrat tu, která je pro daný účel nejlepší.

Tato práce se zabývá podrobným popisem jedné z těchto metod a to konkrétně asociační analýzou, se kterou se setkáme v následující kapitole. Zde je uvedeno pro ucelenější přehled několik jiných metod, které se často používají.

10 Gerstner Laboratory, ČVUT v Praze: Předzpracování dat pro data mining: metody a nástroje. 2006

2.4.1 Klasifikace a predikce

Obecně je klasifikace metodou pro rozdělování dat do skupin dle jistých kriterií. Naproti tomu predikce je postupem, kdy se na základě známé množiny vstupních hodnot a na základě známých jím odpovídajících výstupních hodnot, hledá nejpravděpodobnější hodnota výstupu pro předem neznámé kombinace vstupních hodnot.[11]

Za příklad klasifikace si můžeme uvést firmu, která má prodejny v různých městech a shromažďuje si informace o prodávaných výrobcích. Z těchto informací je možné vytvářet skupiny například podle obratu, podle typu zákazníka či podle sortimentu. Na základě těchto skupin je možné vhodně přizpůsobit reklamní strategii podniku.

Jako příklad predikce nám může posloužit banka, která rozhoduje o přidělování úvěrů. Banka má záznamy o všech poskytnutých úvěrech a z nich je možné zjistit, jací lidé měli problémy se splácením. Pokud tedy přijde do banky zájemce o udělení úvěru, lze díky informacím o něm odhadnout, zda dokáže úvěr splácet.

2.4.2 Shluková analýza

R. E. Bonner definoval v roce 1964 shlukovou analýzu takto: *„Je dána množina objektů, z nichž je každý definován pomocí množiny znaků s ním souvisejících. Tato množina znaků je pro každý objekt stejná. Máme nalézt shluky objektů (podmnožiny původní množiny objektů) tak, aby si členové shluku byli vzájemně podobní, ale nebyli si příliš podobní s objekty mimo tento shluk.“*. Shluková analýza tedy slouží jako prostředek k získání klasifikace.[12]

Shlukovou analýzu v podstatě každý běžně používáme. Například při přebírání ovoce jej rozdělujeme na jakostní a špatné. Nebo při výběru auta si jednotlivé vozy pravděpodobně rozdělíme do shluků podle značky, barvy, kvality apod. S daty v databázích je to podobné. Abychom se v nich lépe vyznali, snažíme se je rozdělit do tříd podle předem definovaného konceptu.

2.4.3 Neuronové sítě

Jedná se v podstatě o analogii našeho lidského mozku, který je tvořen vzájemně propojenými neurony. V umělých sítích se neuron chová jako prvek, který přijímá na vstupech podněty od ostatních neuronů, které jsou s ním propojeny. Pokud celkový součet hodnot na vstupech překročí stanovený práh, neuron se aktivuje a začne ovlivňovat ostatní neurony. Důležitou vlastností těchto sítí je schopnost učení se. Znalosti jsou pak v těchto sítích udržovány díky nastavení vah na jednotlivých vazbách mezi neurony.[13]

---

11 Pulpán, J.: Dolování dat aneb Hledání skrytých souvislostí. Červen 2001. Článek dostupný na URL http://www.systemonline.cz/clanky/dolovani-dat-aneb-hledani-skrytych-souvislosti.htm (březen 2008)

12 Žák, L.: Automatizace. Ročník 47, březen 2004, číslo 3, str. 180

13 Motivováno textem Pospíšil, J., Nemrava, M.: Dolování dat a jeho aplikace. 2006, kapitola 3.4

2.4.4 Rozhodovací stromy

Jedná se o analytické nástroje sloužící k nalezení pravidel a vztahů v datovém souboru pomocí systematického rozdělování a větvení na nižší úrovně. Vzhledem k tomu, že se jedná o stromovou datovou strukturu, tak je algoritmus velice rychlý.

Cílem je určit takové objekty popsané souborem atributů, které dokáží záznamy rozdělit do tříd. Problémem může být určení na kolik „větví“ se má dělit každý objekt. Pokud záznamy rozdělíme do velkého množství tříd, může nastat situace, kdy do každé třídy přísluší pouze několik málo záznamů a nelze tak vyvodit žádná rozhodovací pravidla. [14]

Stromy se skládají z uzlů, které reprezentují testy hodnot určitého atributu objektu. Z uzlu vede konečný počet hran, které jsou množinou hodnot daného atributu a koncové uzly (listy) reprezentují třídy, do kterých je daný objekt klasifikován.[15]

Rozhodovací stromy jsou vhodné pro úlohy, ve kterých má být provedena klasifikace nebo předpověď. Užitečné jsou v oblastech, ve kterých můžeme hodnoty proměnných rozdělit do relativně malého počtu skupin. Na druhou stranu nejsou vhodné pro případy, kdy je úkolem předpovězení kvantitativních hodnot.

Rozhodovací stromy získaly popularitu, protože jejich zobrazení je velmi snadno pochopitelné. Popis pomocí rozhodovacího stromu je řadou jednoduchých rozhodovacích pravidel, které často bývá prezentováno ve formě grafu. Tyto grafy jsou snadno a bez větších znalostí statistických metod interpretovány řídícími pracovníky.[16]

14 Podle článku Rozhodovací stromy, Data mining solutions. Říjen 2002. Článek dostupný na URL http://datamining.xf.cz/view.php?cisloclanku=2002102802

15 Podle člunku Rozhodovací stromy, Wikipedie. Srpen 2007. Článek dostupný na URL http://cs.wikipedia.org/wiki/Rozhodovací_stromy (duben 2008)

16 Půlpán, J.: Dolování dat aneb Hledání skrytých souvislostí. Červen 2001. Článek dostupný na URL http://www.systemonline.cz/clanky/dolovani-dat-aneb-hledani-skrytych-souvislosti.htm (březen 2008)

3 Asociační analýza

Získávání asociačních pravidel hledá zajímavé asociace nebo korelace (míra závislosti mezi dvěma proměnnými) nad velkými množinami datových položek. Nalezení zajímavých asociací nad obchodními transakčními záznamy může pomoci v procesu obchodního rozhodování, jako je návrh katalogů, akčních nabídek nebo rozmístění zboží v obchodě.[17]

Nejčastěji používaným příkladem asociační analýzy je analýza nákupního košíku (market basket analysis). Cílem této analýzy je odhalovat zvyklosti a stereotypy v nákupech zákazníků. Jedná se tedy o hledání produktů, které si zákazníci kupují významně častěji společně. Jako elementární příklad nám může posloužit závislost, že pokud si zákazník kupuje mouku, tak si s vysokou pravděpodobností koupí i cukr. Při rozboru této závislosti si můžeme uvědomit, že pokud si zákazník kupuje mouku, tak ji obvykle upotřebí na pečení, při kterém většinou potřebuje také cukr. Díky odhalení této korelace pak může zareagovat management obchodu a to například rozestavením zboží v obchodě. Výrobky, které se prodávají celkem často společně, pak může umístit blízko u sebe a společný prodej tak posílit. Pokud se dané výrobky prodávají společně ve většině případů, může management zareagovat tak, že tyto výrobky umístí daleko od sebe. Zákazník tak musí projít celým obchodem a může si koupit i jiné zboží, pro které původně nešel. Objevených závislostí se dá také využít při vytváření propagačních letáků nebo při promýšlení slevových akcí.

Využití metod asociační analýzy ale u podpory rozhodování managementu nekončí. Využívá se také při výzkumech v medicíně, sociologii, ale například i v hutnictví.

3.1 Asociační pravidla

Z úvodu již víme, že asociační pravidlo nám reprezentuje korelaci, tedy vzájemný vztah mezi nějakými proměnnými. Asociační pravidlo nám tedy může říkat, že proměnná A se často vyskytuje ve spojení s proměnnou B. Pokud provádíme dolování na obrovském vzorku dat, tak vzájemné porovnání všech prvků v databázi by mohlo trvat neúměrné množství času. Je tedy třeba promyslet, jakým způsobem efektivně vyhledávat asociační pravidla.

Vzájemných vztahů (pravidel) mezi všemi záznamy je možné nalézt prakticky neomezené množství. Proto je třeba stanovit, jakým způsobem určíme, jestli je nalezený vtah relevantní, tedy má pro nás nějakou hodnotu. Hledáme proto vztahy, které se vyskytují v záznamech významně často a kvůli tomu musíme stanovit práh, podle kterého určíme, jestli je daný vztah pro nás zajímavým nebo není. Vždy si ale musíme uvědomovat, že nalezená pravidla jsou pouze hypotézami, které je třeba zkoumáním potvrdit nebo zamítnout.

17 Zendulka, J. a kol., Získávání znalostí z databází, studijní opora. Brno 2006, FIT VUT v Brně s. 75

Z předchozího textu tedy vyplývá, že pro dolování asociačních pravidel vyvstávají dvě základní otázky:

- Jak efektivně vyhledávat závislosti v obrovském množství dat?
- Jak určit, které asociace jsou relevantní?

Odpověď na první otázku nabídne kapitola 4, která pojednává o dolování asociačních pravidel. Následující podkapitoly se snaží osvětlit odpověď na otázku druhou, tedy na určování relevance nalezených asociací.

3.2 Stanovení relevance pravidla

Následující příklady asociačních pravidel se týkají analýzy nákupního košíku. Předpokládejme, že máme množinu všech položek zboží, které obchod nabízí. Každou položku pak můžeme v databázi reprezentovat boolovskou hodnotou, která značí její přítomnost či nepřítomnost v nákupním košíku zákazníka. Záznam o každém nákupu můžeme tedy reprezentovat bitovým vektorem. Z těchto vektorů můžeme pak analýzou získávat asociační pravidla.

Nejprve je třeba ukázat, jak vypadá obecný zápis asociačního pravidla:

$$i_1 \wedge i_2 \wedge i_3 \wedge \ldots \wedge i_{n-1} \Rightarrow in \ (podpora = s, spolehlivost = c)$$

I = $\{i_1, i_2, i_3, i_n\}$ je množina všech položek zboží v databázi. Uvedené pravidlo nám říká, že pokud si zákazník zakoupí všechny položky $i_1 \ldots i_{n-1}$, tak si s podporou s a se spolehlivostí c koupí i položku i_n. Pro lepší pochopitelnost následuje konkrétní příklad:

$$televize \Rightarrow pokojová\ anténa (s = 4\%, c = 75\%)$$

Toto pravidlo ukazuje, že ve 4% z celkového počtu nákupů si zákazníci koupili televizi a zároveň pokojovou anténu. Dále se z pravidla dozvídáme, že pokud si zákazník koupil televizi, tak si v 75% případů koupil i pokojovou anténu.

Proměnná *s* (anglicky support) stanovuje podpor a *c* (anglicky confidence) ukazuje spolehlivost nalezeného pravidla. Tyto dvě proměnné představují základní míru relevance (zajímavosti) asociačního pravidla.

Podpora – jedná se o pravděpodobnost výskytu záznamu, který splňuje všechny predikáty $i_1 \ldots i_n$ v datech. Jde tedy o poměr počtu záznamů obsahujících všechny položky $i_1 \ldots i_n$ ku všem záznamům.

Spolehlivost – jedná se o podmíněnou pravděpodobnost pro jev i_n v datech za podmínky, že platí všechny predikáty $i_1 \ldots i_{n-1}$. Jde tedy o poměr počtu záznamů obsahujících všechny položky $i_1 \ldots i_n$ ku všem záznamům obsahujícím pouze položky $i_1 \ldots i_{n-1}$.

Asociační pravidlo, které překročí uživatelem nastavené prahové hodnoty podpory a spolehlivosti, je považováno za zajímavé a stává se tak součástí výsledku dolování.

3.3 Rozdělení asociačních pravidel

Asociační pravidla se rozdělují mnoha způsoby a podle rozlišných kritérií. Různé druhy pravidel se také často liší i ve způsobu jejich získávání. Následuje přehled některých typů pravidel. Tento přehled vzhledem k rozsáhlosti disciplíny dolování asociačních pravidel není úplný a ukazuje jen několik příkladů.[18]

- **Podle typu hodnot, použitých v pravidlech:**
  - *boolovská pravidla:* nabývají pouze dvou stavů a používají se k určení přítomnosti nebo nepřítomnosti položky v datech.
  - *kvantitativní pravidla:* mohou nabývat spojitých hodnot, a proto se pomocí těchto pravidel dají vyjádřit složitější vztahy. Následující pravidlo může posloužit jako příklad:

$$příjem = 30000 – 40000 \wedge věk = 25 – 30 \Rightarrow koupí\ (notebook)$$

- **Podle typu pravidel:**
  - *implikační:* jedná se o pravidla typu $A \Rightarrow B$, což znamená, že A je příčinnou B.
  - *asociační:* tato pravidla odhalují souvislost mezi daty. Zapisují se $A \sim B$, tedy A souvisí s B.
- **Podle dimenzí obsažených v pravidlech:**
  - *jednodimenzionální pravidla:* jedná se o všechny transakční asociace, které jsou reprezentovány boolovskou hodnotou. Pravidla obsahují pouze jeden predikát, tedy například *koupil*:

$$koupil\ (olivy) \wedge koupil\ (sýr) \Rightarrow koupil\ (víno)$$

  - *multidimenzionální pravidla:* kombinují více predikátů. Příklad:

$$příjem\ (60000 – 75000) \wedge věk\ (45 – 55) \Rightarrow koupil\ (Audi)$$

  - *hybridně dimenzionální pravidla:* pravidla v nichž se některé predikáty opakují. Příklad:

$$věk\ (19 – 25) \wedge zaměstnání\ (student) \wedge koupil\ (notebook) \Rightarrow koupil\ (tiskárna)$$

- **Podle úrovní abstrakce použité v pravidlech:**

  Při hledání asociací můžeme také rozlišovat úroveň abstrakce. Můžeme například zkoumat vliv věku zákazníka na výběr buď značky auta nebo jen na typ vozu (sedan, kombi). První příklad je konkrétnější.
- **Podle dalších rozšíření asociačních pravidel:**

  Hledání asociací může být obohaceno o další rozšíření, například o korelační analýzu, o další statistické testy při vyhodnocování zajímavosti pravidel, atd.

18 Podle zdrojů Burda, M.: Získávání znalostí z databází – Asociační pravidla. VŠB-TU Ostrava, 2004 a Zendulka, J. a kol., Získávání znalostí z databází, studijní opora. Brno 2006, FIT VUT v Brně

4 Dolování asociačních pravidel

Kniha Data mining: Concepts and Techniques (J. Han, M. Kamber) označuje za autory dolování asociačních pravidel skupinu tří badatelů Agrawal, Imielinski a Swami, kteří v roce 1993 publikovali článek *„Mining association rules between sets of items in large databases“*. Zajímavostí ovšem je, že se touto disciplínou o několik let dříve zabývala skupina českých vědců a to pánové P. Hájek, M. K. Chytil a T. Havránek, kteří společně navrhli metodu GUHA (General Unary Hypothesis Automaton, česky automat na obecné unární hypotézy). Tato metoda sloužila pro systematické vytváření hypotéz na základě empirických dat, tedy umožňovala generování pravidel a to asociačních, korelačních či implikačních. Rozbor této metody je ovšem nad rámec této práce.

Tato kapitola vysvětluje základní myšlenku dolování asociačních pravidel. Dále se čtenář seznámí s principy triviálního vyhledávání, s algoritmem Apriori a s jeho modifikacemi. Algoritmu Apriori je věnováno hodně prostoru a to především proto, že je v oboru dolování asociačních pravidel považován za základní.

4.1 Úvod do problematiky

V následujícím textu se budeme podrobněji zabývat již zmiňovanou analýzou nákupního košíku, což znamená dolováním jednodimenzionálních asociačních pravidel. Jednodimenzionální pravidla obsahují pouze jeden predikát, a to v tomto případě predikát *koupil*.

Dolování budeme provádět na transakční databázi, která zaznamenává detailní informace o nákupech zákazníků. Analyzovaná data se tak budou skládat z řady boolovských atributů (položky zboží v obchodě), které obsahují hodnotu 1 pokud si zákazník danou položku koupil a hodnotu 0 pokud ne (viz. tabulka).

TID	položka 1	položka 2	položka 3	položka 4	položka 5
01	**1**	**1**	0	0	**1**
02	0	0	0	1	0
03	1	1	1	0	0
04	**1**	**1**	1	0	**1**
05	0	1	0	1	1

Tabulka 1: Databáze transakcí

Z této tabulky můžeme získat toto pravidlo:

$$položka1 \wedge položka2 \Rightarrow položka5$$

Získané pravidlo má podporu 2/5 = 0,4 = 40% a spolehlivost 2/3 = 0,67 = 67%. Pokud bychom nastavili prahové hodnoty pro podporu na 20% a pro spolehlivost na 80%, tak bychom toto pravidlo prohlásili za nezajímavé.

Proces dolování asociačních pravidel probíhá ve dvou krocích:

- *Nalezení frekventovaných množin* – množina se nazývá frekventovanou, jestliže dosahuje minimální stanovené podpory.
- *Generování silných asociačních pravidel* – asociační pravidlo se nazývá silné tehdy, jestliže je vygenerováno z frekventované množiny a spolehlivost pravidla dosahuje minimální stanovené hodnoty.

4.2 Triviální vyhledávání

Na základě předchozího popisu již můžeme navrhnout nejjednodušší algoritmus vyhledávání pravidel. Ten by pracoval tak, že by generoval kombinace predikátů na pravou i levou stranu pravidla a testovali bychom, jestli se jedná o silné asociační pravidlo.

Tento algoritmus je v praxi nepoužitelný, protože není nijak omezen počet nabízených položek (atributů), ani počet záznamů v databázi (transakcí). Takto navržený algoritmus má exponenciální časovou složitost a při vyhledávání pravidel v obrovském množství dat dojde ke generování neomezeného počtu kombinací.

4.3 Algoritmus Apriori

Apriori je nejjednodušším a často používaným algoritmem pro dolování asociací. Samozřejmě již není tak jednoduchý jako předchozí princip triviálního vyhledávání. Při generování frekventovaných množin využívá jisté apriori znalosti (apriori = předem známé), která algoritmu pomáhá zabránit generování nekonečných kombinací.

Tato apriori znalost říká, že každá podmnožina frekventované množiny musí být také frekventovaná. Vycházíme přitom z faktu, že přidání nového prvku k množině nikdy nemůže zapříčinit nárůst podpory této množiny. Pokud tedy nějaká nalezená množina není frekventovaná, tak jí algoritmus při dalším prohledávání vyloučí. Generování frekventovaných množin $k+1$ položek se tedy provádí na frekventovaných množinách obsahujících k položek.

Apriori algoritmus funguje tak, že nejprve vyhledá všechny frekventované množiny obsahující jednu položku (kardinalita 1). Množinu frekventovaných množin kardinality 1 nazveme L_1. V dalším kroku jsou vyhledávány frekventované množiny kardinality 2 (L_2). K jejich vyhledávání je využito množiny L_1. Pro nalezení množiny L_3 je využito množiny L_2, atd. Algoritmus ukončí vyhledávání ve chvíli, kdy již nelze najít frekventovanou množinu vyšší kardinality. Pro nalezení frekventovaných množin každého stupně je třeba jeden celý průchod daty.

Pro bližší popis generování množiny L_k z množiny L_{k-1} je třeba ještě definovat pojem *kandidát na frekventovanou množinu*. Jde o množinu u níž ještě frekventovanost nebyla prokázána průchodem daty, ale zároveň nebyla prohlášena za nefrekventovanou.

4.3.1 Dva kroky algoritmu

Generování frekventované množiny kardinality *k* probíhá ve dvou krocích, které popisuje tato podkapitola.

Spojovací krok: pro nalezení všech frekventovaných množin kardinality *k* (L_k), se spojením množin z L_{k-1} vygeneruje množina kandidátů na frekventované množiny (C_k). Algoritmus předpokládá lexikografické seřazení položek v množině. Ke spojení dvou množin l_i a l_j (množiny z L_{k-1}) dojde, pokud mají tyto množiny prvních *k-2* prvků shodných. Pro množiny l_i a l_j dále musí platit, že (*k-1*) – položka je u množiny l_i lexikograficky menší než u množiny l_j. Výsledný kandidát tedy vznikne spojením l_i s poslední (*k-1*) položkou l_j.

Vylučovací krok: Jestliže máme vygenerovanou množinu C_k (nadmnožina L_k), tak z ní v tomto kroku musíme vyloučit všechny množiny, které nejsou frekventované. O frekventovanosti množin se lze přesvědčit jedním průchodem databází, což je ovšem u velkých databází velmi pomalé a neefektivní. Proto při redukování využíváme Apriori znalosti. Z C_k tedy vyloučíme ty množiny, pro které neplatí, že všechny jejich podmnožiny jsou frekventované. Pro porovnávání se dá využít hashovací strom, což výrazně urychlí výsledné prohledávání. S takto redukovanou množinou C_k provedeme průchod daty a ověříme splnění minimální stanovené podpory u všech zbylých kandidátních množin. Poté již získáváme hledanou množinu L_k.

4.3.2 Funkce algoritmu

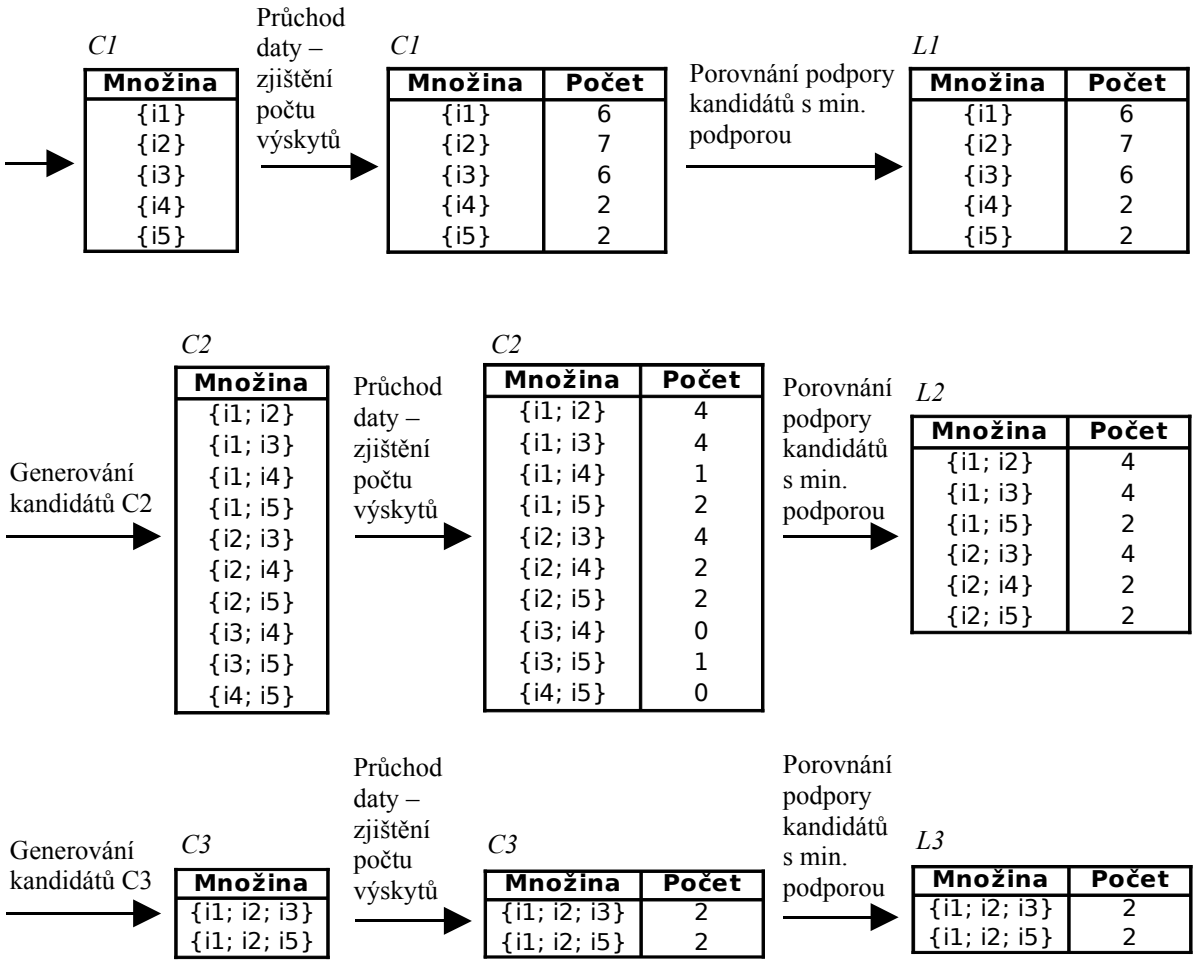
Pro lepší pochopení celého principu je v této podkapitole ukázána funkce algoritmu Apriori na jednoduchém příkladu.[19]

TID	položky
01	i1, i2, i5
02	i2, i4
03	i2, i3
04	i1, i2, i4
05	i1, i3
06	i2, i3
07	i1, i3
08	i1, i2, i3, i5
09	i1, i2, i3

Tabulka 2: Transakční data

19 Uvedený příklad je částečně převzat ze studijní opory: Zendulka, J. a kol., Získávání znalostí z databází, studijní opora. Brno 2006, FIT VUT v Brně s. 78 - 80

- Nejprve vygenerujeme množinu všech kandidátů na frekventované množiny kardinality jedna (C_1). Následuje průchod daty, při kterém se zjišťuje počet výskytů každé kandidátské množiny. Pro každou množinu v C_1 se vypočítá podpora.
- Ve druhém kroku dochází k vyloučení všech kandidátních množin, které nesplňují minimální stanovenou podporu. Takto vytvoříme množinu frekventovaných množin kardinality jedna (L_1). V příkladu budeme uvažovat minimální podporu 22%.
- Ve třetím kroku se vzájemným kombinováním položek z L_1 vytváří množina kandidátů na frekventované množiny kardinality dva (C_2). Poté je opět jedním průchodem daty vypočítána podpora jednotlivých kandidátů. Z množiny C_2 jsou vyloučeny všechny kandidátní množiny, které nesplňují stanovenou prahovou hodnotu podpory a je tak vygenerována množina L_2.
- Algoritmus pokračuje generováním další množiny kandidátů C_3 a z ní je výše popsaným způsobem vygenerována množina L_3.
- Algoritmus skončí ve chvíli, kdy po odstranění všech nefrekventovaných kandidátních množin zůstane množina C_k prázdná, tedy nezůstane žádná frekventovaná množina kardinality *k*.



Obrázek 3: Ukázka průběhu algoritmu Apriori

Pokud jsme již nalezli všechny frekventované množiny v datech, tak vygenerování silných asociačních pravidel již není složité. Probíhá podle následující rovnice:

$$spolehlivost(A \Rightarrow B) = P(B|A) = \frac{s(A \cup B)}{s(A)}$$

Samotné generování asociačních pravidel pak probíhá v následujících dvou krocích:

- Pro každou frekventovanou množinu položek *l* se vygenerují všechny neprázdné podmnožiny.
- Pro každou neprázdnou podmnožinu s množiny l se vygeneruje asociační pravidlo ve tvaru $s \Rightarrow (l - s)$ a podle rovnice se vypočítá jeho spolehlivost. Jestliže je spolehlivost pravidla vyšší než minimální stanovená , pak je toto pravidlo silné.

Vzhledem k tomu, že jsou silná asociační pravidla generována z frekventovaných množin, je samozřejmé, že splňují podmínku minimální stanovené podpory.

Pro úplné dokončení příkladu funkce algoritmu Apriori, následuje ukázka vygenerování asociačních pravidel pro množinu *{i1, i2, i5}*. Neprázdnými podmnožinami jsou *{i1, i2}*, *{i1, i5}*, *{i2, i5}*, *{i1}*, *{i2}*, *{i5}*. Vygenerují se tedy následující asociační pravidla:

$i2 \wedge i5 \Rightarrow i1$; *spolehlivost = 2/2 = 100%* $\quad$ $i1 \wedge i5 \Rightarrow i2$; *spolehlivost = 2/2 = 100%*

$i2 \Rightarrow i1 \wedge i5$; *spolehlivost = 2/7 = 29%* $\quad$ $i5 \Rightarrow i1 \wedge i2$; *spolehlivost = 2/2 = 100%*

$i1 \wedge i2 \Rightarrow i5$; *spolehlivost = 2/4 = 50%* $\quad$ $i1 \Rightarrow i2 \wedge i5$; *spolehlivost = 2/6 = 33%*

4.3.3 Zvýšení efektivity algoritmu Apriori

- **Hashování:** Při generování kandidátů kardinality dva (C2) z frekventovaných množin L1, lze použít hashovací tabulku. Vygenerujeme všechny kombinace množin L1 a ukládáme je do hashovací tabulky. Využijeme při tom faktu, že pro identické záznamy je vygenerována stejná hash hodnota a tak jsou tyto záznamy zařazeny do stejného seznamu v tabulce. U každého seznamu tabulky je záznam o počtu prvků v něm obsažených. Pokud je počet prvků v některém seznamu menší než minimální stanovená podpora, tak víme, že všechny množiny v něm obsažené jsou nefrekventované. Proto se s tímto seznamem již dále nepočítá, což vede k úspoře časové náročnosti algoritmu.
- **Rozdělení dat:** Nejprve se databáze rozdělí na *n* částí. Velikost *n* se volí tak, aby se všechna data vešla do paměti. V jednotlivých částech se pak hledají *lokální frekventované množiny*. Tyto potenciálně frekventované množiny se stanou kandidáty na získání globálních frekventovaných množin. Výhodou této metody je, že pro nalezení frekventovaných množin jsou potřeba jen dva průchody daty.
- **Redukce transakcí:** Pokud transakce neobsahuje žádnou frekventovanou množinu kardinality *k*, tak nemůže obsahovat ani frekventovanou množinu kardinality *k+1*. Proto je

vhodné takové transakce z databáze buď vymazat nebo vhodně označit, aby se s nimi již nepočítalo.

- **Vzorkování:** Jedná se o velmi rychlou a efektivní metodu. Nejprve se náhodně vybere vzorek transakcí z databáze tak, že se celý vzorek uloží do paměti. Poté se v něm vyhledávají frekventované množiny. To vede k jisté nepřesnosti, proto se počítá s menší minimální podporou. Díky tomu získáme i některé množiny, které nejsou frekventované. Proto nakonec ověříme frekventovanost všech potenciálně frekventovaných množin jedním průchodem celou databází.[20]

4.3.4 Modifikace algoritmu Apriori

Existuje několik modifikací algoritmu Apriori, které se v praxi používají. Pro přehled jsou zde uvedeny některé z nich. Podrobnější popis těchto algoritmů je ovšem nad rámec této práce.

- **AprioriItemset:** hlavní odlišnost algoritmu spočívá v odlišné reprezentaci výchozí informace, která umožňuje efektivnější provádění některých operací.[21]
- **AprioriTID:** tento algoritmus nepracuje pouze s fyzickou databází na disku jako původní Apriori. Využívá množinu uloženou v paměti, která je obrazem databáze. Značnou výhodou je snižování velikosti množiny v paměti, jelikož prvky, které nemohou být vhodnými kandidáty, vůbec neobsahuje.
- **AprioriTIDList:** algoritmus se snaží využít výhod obou předešlých a potlačit jejich nevýhody.[22]

4.4 Shrnutí

V procesu získávání asociačních pravidel je vždy nejdůležitější rozhodnout, která pravidla jsou pro nás zajímavá. Jsou pro to využívány různé statistické postupy či testy. Je třeba si ale uvědomit, že statistika pracuje především s pravděpodobností, a proto vždy existuje jistá možnost (byť velmi malá), že statistické závěry budou nepravdivé. Je tedy pravděpodobné, že mezi získanými asociačními pravidly bude alespoň některé chybné.

Získávání asociačních pravidel tím ale svůj smysl neztrácí. Na získané pravidla nemůžeme pohlížet jako na jistou pravdu. Asociační pravidla nám ukazují, které hypotézy jsou danými daty podporovány, jaké vztahy jsou nejspíše pravdivé a které směry dalšího bádání jsou nadějné.[23]

20 Podle zdrojů Burda, M.: Získávání znalostí z databází – Asociační pravidla. VŠB-TU Ostrava, 2004 a Zendulka, J. a kol., Získávání znalostí z databází, studijní opora. Brno 2006, FIT VUT v Brně

21 Honzík, M. J.: Výzkum informačních a řídících systémů. Brno, VUT FEI, 1999

22 Zeman, D.: Description of association rules discovery, process in image databases. [Semestrální práce] Brno, VUT FIT

23 Motivováno zdrojem: Burda, M.: Získávání znalostí z databází – Asociační pravidla. VŠB-TU Ostrava, 2004

5 Praktická část

5.1 Uvedení

Praktickou částí této práce je implementace aplikace, která vhodným způsobem prezentuje funkčnost algoritmu Apriori. Aplikace je implementována v programovacím jazyce Java.

Jedná se o konzolovou aplikaci, která jako vstupní databázi používá soubor zapsaný dle daného formátu. Soubor obsahuje seznam položek a databázi transakcí. Program nejprve načte položky a poté v databázi transakcí pomocí algoritmu Apriori vyhledává závislosti mezi položkami. Po jejich nalezení program vygeneruje ukázková asociační pravidla a do uživatelem definovaného umístění vytvoří HTML soubor, který prezentuje výsledky průběhu algoritmu.

5.2 Návrh aplikace

Aplikace je vytvořena ze třech základních tříd. Hlavní třídou je Main.java, která obsahuje funkci main(). Tato třída obstarává komunikaci s uživatelem, od kterého si vyžádá veškeré vstupní informace. Další třídou je Loader.java. Tato třída vytváří abstrakci nad čtenými daty, tedy obstarává veškeré čtení ze souboru i ze standardního vstupu. Data předává v dohodnutém formátu. Poslední ze základních tříd je Apriori.java. Třída obsahuje algoritmus Apriori, vyhledává závislosti v uložených datech a generuje asociační pravidla.

Program dále obsahuje dvě pomocné třídy, které slouží pro uložení generovaných množin a pro uložení výsledných asociačních pravidel.

Poslední třída, kterou aplikace obsahuje je Htmlgenerator.java, který slouží pro generování HTML souboru prezentujícího průběh algoritmu a zobrazujícího nalezené asociace.

5.2.1 Main.java

Třída obsahuje funkci main(), která se volá při spuštění programu. Nejprve se zkontrolují parametry příkazové řádky. Poté program komunikuje pomocí metod třídy Loader.java přes standardní vstup a výstup s uživatelem a vyžádá si od něj základní informace jako je název a cesta k souboru s databází a parametry pro vyhledávání asociačních pravidel, tedy minimální hodnoty podpory a spolehlivosti. Jakmile jsou zadány veškeré vstupní údaje spustí se třída Apriori.java. Po jejím průběhu se volá generátor HTML souboru prezentujícího výsledný průběh algoritmu.

Třída také ošetřuje všechny chybové stavy. Pokud při běhu programu dojde k zadání neplatného vstupu nebo je generována výjimka, volá se funkce printErrorExit(), která vytiskne na standardní výstup chybové hlášení a ukončí běh programu.

5.2.2 Loader.java

Tato třída obsahuje několik metod, které slouží pro komunikaci s uživatelem. Jedná se o metody questionString(), questionInteger() a questionTF(). Každá z těchto metod má za parametr řetězec, který je jako otázka vytištěn na standardní výstup. Poté je od uživatele vyžadována syntakticky správná odpověď (například questionInteger() vyžaduje jako vstup pouze číselné hodnoty, jinak vypíše chybové hlášení a vyžaduje opětovné zadání hodnoty). Načtenou a syntakticky zkontrolovanou odpověď pak metody předávají jako svou návratovou hodnotu.

Další důležitou funkcí této třídy je také to, že se stará o načítání dat ze souboru obsahujícího databázi položek a transakcí (podrobný popis formátu souboru je v následující kapitole 6.3.2). Tuto funkci ve třídě obstarávají dvě metody a to loadItems() a loadTransa(). První z těchto metod načte ze souboru seznam položek, vytvoří pole se všemi indexy a také tabulku, která umožňuje převod indexu položky na její název. Druhá z metod funguje tak, že při volání vrátí jako návratovou hodnotu jednu transakci. Jakmile metoda přečte všechny transakce a narazí na konec souboru, vrátí prázdný řetězec a soubor uzavře.

Poslední metodou, kterou tato třída obsahuje, je generateTransaFile(). Tato metoda umožňuje vygenerovat soubor s testovacími daty.

5.2.3 Apriori.java

Jedná se o nejdůležitější třídu celé aplikace. Apriori.java obsahuje algoritmus Apriori a veškeré funkce, které jsou pro jeho průběh nezbytné.

Nejprve je třeba nastavit některé parametry třídy, které jsou nutné pro samotný průběh algoritmu. Jedná se o seznam položek v databázovém souboru, který je nutné pro generování první kandidátské množiny. Dále pak je třeba nastavit minimální hodnotu podpory a vzhledem k tomu, že třída generuje i asociační pravidla, tak se musí zadati i prahová hodnota spolehlivosti. Po nastavení těchto hodnot se již volá metoda process(), která řídí průběh algoritmu. Metodě se jako parametr předává odkaz na databázový soubor.

```
C1 = generateCandidate()    //generování kandidátů C1
executionSupport(C1)        //výpočet podpory kandidátů
L1 = checkSupport(C1)       //vyloučení nefrekventovaných
                            //kandidátů

while(počet kandidátů Ln != 0) {
     Cn = generateCandidate()
     executionSupport(Cn)   //průchod všech transakcí
                            //a výpočet výskytů v Cn
     Ln = checkSupport(Cn)  //vyloučení množin, které
                            //nejsou frekventované
}
```

Ilustrace 1: Ukázka principu algoritmu Apriori v pseudokodu

Algoritmus Apriori probíhá v krocích, které jsou znázorněny na předchozí straně v pseudokódu (*Ilustrace 1*). Pro každou kandidátskou množinu C_n algoritmus projde jednou celou databázi a ověří podporu všech jejich prvků. Vyloučením prvků s nízkou podporou pak vznikne frekventovaná množina L_n. Všechny kandidátské i frekventované množiny jsou v průběhu ukládány do vektorů, aby bylo možné znázornit průběh algoritmu ve výsledném HTML souboru, který prezentuje výsledky.

Třída Apriori.java obsahuje především pomocné metody pro průběh algoritmu. Zahrnuje v sobě také ještě metody pro generování výsledných asociačních pravidel. Hlavní funkci při tom zastává metoda generateAssociationRules(), která projde všechny frekventované množiny nejvyšší nalezené kardinality a vygeneruje pro ně silná asociační pravidla tedy ta, která splňují požadavek minimální stanovené spolehlivosti. Vzhledem k tomu, že se asociační pravidla generují ze všech podmnožin frekventované množiny, tak třída obsahuje důležitou pomocnou metodu getAllSubsets(), která generuje všechny neprázdné podmnožiny z množiny předané parametrem.

5.2.4 Třídy pro uložení dat

Program obsahuje dvě třídy, které jsou v podstatě využívány jako datové struktury. AprioriSets.java slouží pro uložení množiny položek. Při generování kandidátských množin se vytvoří instance této třídy a uloží se do ní samotná kandidátská množina, dále množina, ze které daná kandidátská množina vznikla (je třeba při generování dalších kandidátů) a nakonec se nastaví počet výskytů množiny na nula. Při průchodu databází se vyhledává daná množina v transakcích a případně se zvyšuje počet výskytů množiny.

Další třídou je AssociationRules.java, která slouží pro uložení asociačních pravidel a jejich spolehlivosti, které jsou vygenerovány z některé frekventované množiny.

5.2.5 Htmlgenerator.java

Jedná se o třídu, která vygeneruje HTML soubor obsahující informace o průběhu algoritmu. Soubor tedy obsahuje například hodnoty parametrů, se kterými byl program spuštěn, přehled položek v databázi, dále pak ukázky vygenerovaných asociačních pravidel a také jsou v něm ukázky kandidátských a frekventovaných množin.

5.3 Práce s programem

5.3.1 Uživatelem zadávaná data

Program se po svém spuštění nejprve dotáže, zda si uživatel přeje načíst stávající soubor s databází transakcí nebo si přeje vygenerovat nový pro potřeby testování. V případě spuštění generátoru nového databázového souboru je třeba zadat celkový počet položek a transakcí. Dále se program ptá na

konkrétní názvy položek, které ovšem není nutné zadat (za názvy položek se pak použijí identifikátory transakcí). Generátor souboru vytváří náhodně databázi transakcí. Počet výskytů jednotlivých položek je ovšem v takto generovaných transakcích celkem vysoký, a proto tyto databáze obsahují velké množství frekventovaných množin, čímž se výrazně liší od reálných obchodních dat. Programem vytvářené databáze se hodí především pro potřeby testování a také umožňují měření časové náročnosti algoritmu.

Jakmile program načte či vygeneruje platný databázový soubor, tak se ještě dotáže na prahové hodnoty podpory a spolehlivosti, které jsou nutné pro samotný průběh algoritmu Apriori a pro následné generování asociačních pravidel. Po zadání těchto hodnot se spustí algoritmus. Po jeho skončení si ještě program vyžádá zadání názvu pro HTML soubor s výsledky.

5.3.2 Formát vstupních dat

Program používá vlastní formát souboru, který si pro potřeby testování může uživatel lehce sám vytvořit. Soubor je rozdělen do dvou částí, přičemž první obsahuje seznam identifikátorů a názvů položek a druhá část je již samotná databáze transakcí.

```
*Řádkový komentář
[itemList]
i1 = chléb
i2 = rohlíky
i3 = máslo
i4 = jogurt
i5 = šunka

[transactionDatabase]
i1 i2 i5
i2 i4
i2 i3
i1 i2 i4
i1 i3
i2 i3
i1 i3
i1 i2 i3 i5
i1 i2 i3
```

Ilustrace 2: Ukázka databázového souboru [24]

5.3.3 Prezentace výsledků algoritmu Apriori

Program po svém průběhu vytvoří HTML soubor prezentující výsledky a samotný průběh algoritmu. Tento soubor obsahuje přehled vstupních parametrů (podporu, spolehlivost, počet položek v databázi a počet transakcí), dále obsahuje seznam databázových položek a jejich názvů, vygenerovaná asociační pravidla a nakonec přehled kandidátských a frekventovaných množin.

24 Jedná se o databázi použitou v příkladu v kapitole 4.3.2

5.4 Testování programu

Vzhledem k tomu, že je program implementován v programovacím jazyce Java, je aplikace nezávislá na typu operačního systému. Testování programu bylo prováděno jak na MS Windows, tak na OS Linux. Zátěžovými testy byla prokázána přibližně shodná výkonnost programu na obou zmiňovaných platformách.

5.4.1 Ověření funkčnosti

Prvním testem programu, pomocí kterého byla ověřena jeho správná funkčnost, byl test na ukázkové databázi z kapitoly 4.3.2. Výstup programu s výsledky tohoto testu je možné najít v příloze 1. Z této ukázky je patrné, že algoritmus proběhl stejně, jako je uvedeno na obrázku ilustrujícím průběh algoritmu Apriori rovněž v kapitole 4.3.2.

Hlavním testem pro ověření funkčnosti programu byla analýza vzorku databáze z obchodního domu Globus, která byla pro potřeby testování poskytnuta Ing. Vladimírem Bartíkem, Ph.D. Jedná se o kompletní databázi položek nabízený obchodním domem (čítá 62 154 položek), ale databáze transakcí je pouze velmi malým vzorkem čítajícím pouhých 274 transakcí. Analýzou této databáze byla nalezena následující asociační pravidla:

Čaj lemon => Čaj (podpora: 6%, spolehlivost: 53%)
Čaj ovocná40 => Čaj (podpora: 4%, spolehlivost: 66%)
Čaj maracuja => Čaj (podpora: 4%, spolehlivost: 64%)
Čaj orange => Čaj (podpora: 4%, spolehlivost: 76%)

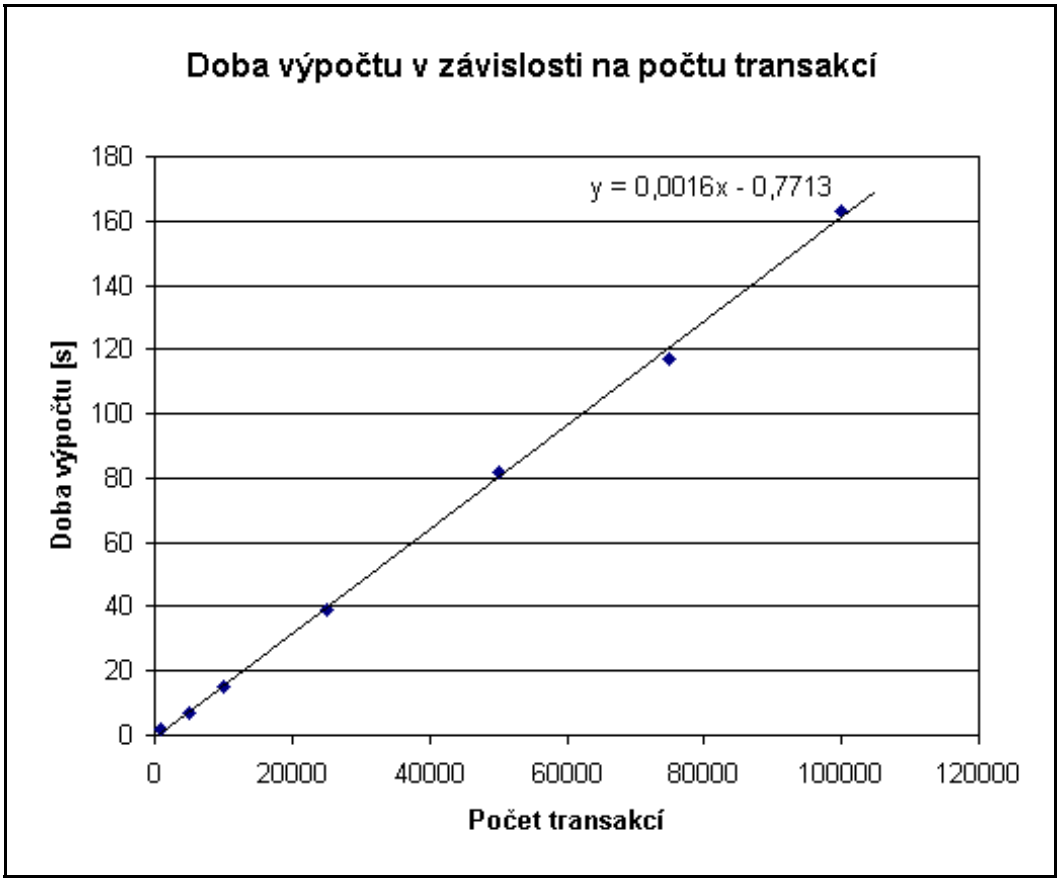
Pro databázi obchodního domu s potravinami se jedná o neočekávaný výsledek. Obvykle se v nejsilnějším pravidle vyskytují komodity z oblasti pečiva, protože ty jsou skutečně prodávány nejčastěji. Fakt, že nebyl nalezen očekávaný výsledek, lze vysvětlit tím, že se jedná o příliš malý vzorek databáze. Skutečnost, že je toto pravidlo v tomto vzorku nejsilnějším, byla potvrzena i jinými metodami, a proto je lze prohlásit za správné.

Test skutečné operační databáze z obchodu také prokázal jistou odlišnost této reálné databáze od programem náhodně generovaných souborů transakcí. Při analýze transakcí z Globusu byl program spuštěn s podporou 3% a spolehlivostí 50%. Při tomto nastavení bylo z celkového počtu položek v databázi nalezeno pouze 51 frekventovaných pro kardinalitu 1, což je necelé jedno promile. Pro kardinalitu 2 bylo nalezeno 1275 kandidátských množin, frekventovaných bylo pouze 17 a z těchto bylo vygenerováno celkem 6 silných asociačních pravidel. U programem generovaných transakčních databází splňují požadavek podpory 20% všechny položky. Obvykle bývají s touto podporou nalezeny i množiny kardinality osm a vyšší. Proto tyto generované databáze nelze srovnávat s reálnými a lze je využít především k zátěžovým testům, které jsou popsány v následující kapitole.

5.4.2 Testy doby výpočtu algoritmu

Tyto testy byly prováděny na programem náhodně generovaných databázích. Při tomto generování se využívá funkce Random() z balíku java.util. Tato funkce vrací hodnoty v rovnoměrném rozložení pravděpodobnosti. Dá se tedy předpokládat, že analýza dvou generovaných souborů se shodným počtem transakcí i položek bude trvat přibližně stejně dlouho. Toto tvrzení bylo ověřeno na analýze třech různých vygenerovaných souborů, které obsahovaly 15 položek a 10 000 transakcí. Analýza trvala u prvního souboru 15 vteřin, u druhého 17 vteřin a u posledního 14 vteřin. Z toho lze usuzovat, že pomocí takto generovaných souborů lze celkem přesně testovat časovou náročnost algoritmu v závislosti na počtu transakcí či na počtu položek v databázi.

Nejprve bude popsán výpočetně jednodušší test, který znázorňuje časovou složitost algoritmu v závislosti na počtu transakcí v databázi. Pro tento test bylo vygenerováno několik pokusných databází vždy s 15 položkami a s rozdílným počtem transakcí. Program byl pro každý test spouštěn se stejnou hodnotou podpory i spolehlivosti a to s podporou 20% a spolehlivostí 70%. Informace o průběhu algoritmu znázorňuje následující graf:



Z výsledků provedeného testu, zobrazených v grafu je vidět, že závislost doby výpočtu algoritmu na počtu transakcí je lineární. V grafu se objevují jen malé odchylky od lineární regresní křivky, které jsou pravděpodobně způsobeny náhodným generováním databáze transakcí.

Druhý test byl výpočetně náročnější. Jedná se o test závislosti doby výpočtu na počtu položek v databázi. Opět bylo programem vygenerováno několik pokusných databází, tentokrát se shodným počtem transakcí, ale odlišným počtem položek. Program byl spouštěn se shodnými parametry jako v předchozím testu. Dobu výpočtu znázorňuje následující graf:

Z grafu můžeme vidět, že závislost doby výpočtu na počtu položek v databázi je přibližně exponenciální. Pokud je počet položek v databázi menší než 20, je doba výpočtu prakticky zanedbatelná. Od 20 položek ovšem doba výpočtu výrazně narůstá. Problémem je, že stanovenou podporu splňují množiny velmi vysokých kardinalit. Zajímavým faktem je, že při testu s 20 položkami bylo nalezeno 364 frekventovaných množin kardinality 5, zatímco při testu s 25 položkami jich bylo již 2128. K časově velmi náročným částem algoritmu Apriori je generování kandidátských množin, které má kvadratickou složitost. Vysoké počty frekventovaných množin kardinality *k* tak vedou k velkému množství kandidátských množin kardinality *k+1*, což algoritmus výrazně zpomalí.

Jak již ale bylo zmíněno v kapitole 5.4.1, tak ve skutečných podnikových databázích se nachází frekventovaných množin jen velmi málo. Pro tyto databáze se dá považovat za obecně platnou spíše časová náročnost algoritmu v závislosti na počtu transakcí v databázi. Z toho lze usuzovat, že čas výpočtu algoritmu je dán počtem položek v databázi a množstvím nacházených frekventovaných množin jednotlivých kardinalit.

6 Závěr

Práce se zabývá dolováním asociačních pravidel pomocí algoritmu Apriori. Teoretická část práce vysvětluje základní principy získávání znalostí z dat a dále se věnuje asociační analýze. Hlavní důraz je zde kladen na vysvětlení principů algoritmu Apriori. Také jsou zde nastíněny možnosti zvýšení jeho efektivity a přehled modifikací tohoto algoritmu.

V rámci praktické části práce byl vytvořen program demonstrující činnost zmiňovaného algoritmu Apriori. Aplikace byla vytvořena v programovacím jazyce Java, díky čemuž je nezávislá na platformě operačního systému. Ověření funkčnosti aplikace bylo provedeno na vzorku podnikové databáze obchodního domu Globus. Nalezené výsledky se shodovaly s výsledky analýz, které byly na tomto vzorku provedeny již dříve. Tím bylo prokázáno, že aplikace funguje správně. Dále bylo pomocí aplikace provedeno několik testů, které se snažily odhalit závislost ve vstupních datech na době výpočtu algoritmu. Bylo zjištěno, že doba výpočtu lineárně stoupá se zvyšujícím se počtem transakcí v databázi. Dále bylo prokázáno, že doba výpočtu výrazně závisí na počtu frekventovaných množin v datech.

Jako možné rozšíření do budoucna se nabízí implementace některých metod pro zvýšení efektivity algoritmu Apriori. Také v současné době existuje několik modifikací tohoto algoritmu, jejichž použití by mohlo vést k výraznému snížení doby výpočtu algoritmu. Další možností rozšíření by mohlo být interaktivní rozhraní pro prezentaci výsledků průběhu algoritmu, které by nabízelo uživateli přesně ty informace, které chce vidět.

Literatura

[1] Han, J., Kamber, M., Data mining: Concepts and Techniques. USA, Academic Press 2001

[2] Lacko, L., Databáze: datové sklady, OLAP a dolování dat s příklady v Microsoft SQL Serveru a Oracle. Brno, Computer Press 2003

[3] Zendulka, J. a kol., Získávání znalostí z databází, studijní opora. Brno 2006, FIT VUT v Brně

[4] Hruška, T.: Informační systémy, pokročilé informační systémy. Brno 2006, studijní opora, FIT VUT v Brně.

[5] Pospíšil, J., Nemrava, M.: Dolování dat a jeho aplikace. 2006

[6] Burda, M.: Získávání znalostí z databází – Asociační pravidla. VŠB-TU Ostrava, 2004

[7] Skolková, L.: Dobývání znalostí z databází. Praha, 2004

[8] Internetová encyklopedie Wikipedia. URL http://www.wikipedia.org/ (březen 2008)

[9] Kocan, M.: Dolujeme a dolujeme. Červen 2006. Článek dostupný na URL http://www.dbsvet.cz/view.php?cisloclanku=2006060101 (březen 2008)

[10] Gerstner Laboratory, ČVUT v Praze: Předzpracování dat pro data mining: metody a nástroje. Praha, 2006

[11] Two Crows Corporation, Potomac USA: Introduction to Data Mining and Knowledge Discovery. 2005

[12] Půlpán, J.: Dolování dat aneb Hledání skrytých souvislostí. Červen 2001. Článek dostupný na URL http://www.systemonline.cz/clanky/dolovani-dat-aneb-hledani-skrytych-souvislosti.htm (březen 2008)

[13] Prokeš, M.: Umíte využít svá data? Vše, co jste chtěli vědět data miningu, ale báli jste se zeptat. Září 2000. Článek dostupný na URL http://www.systemonline.cz/clanky/umite-vyuzit-sva-data.htm (2008)

[14] Žák, L.: Automatizace. Ročník 47, březen 2004, číslo 3, str. 180

[15] Šarmanová, J.: Metody dolování znalostí z dat. Datakon Brno 2002

[16] Honzík, M. J.: Výzkum informačních a řídících systémů. Brno, VUT FEI, 1999

[17] Zeman, D.: Description of association rules discovery, process in image databases. [Semestrální práce] Brno, VUT FIT

[18] Cowley, S.: Dolování dat. 2005. Článek dostupný na URL http://archiv.computerworld.cz/-cwarchiv.nsf/clanky/5C3E3B4604AE67AFC125711700539FEA?OpenDocument (2008)

Seznam příloh

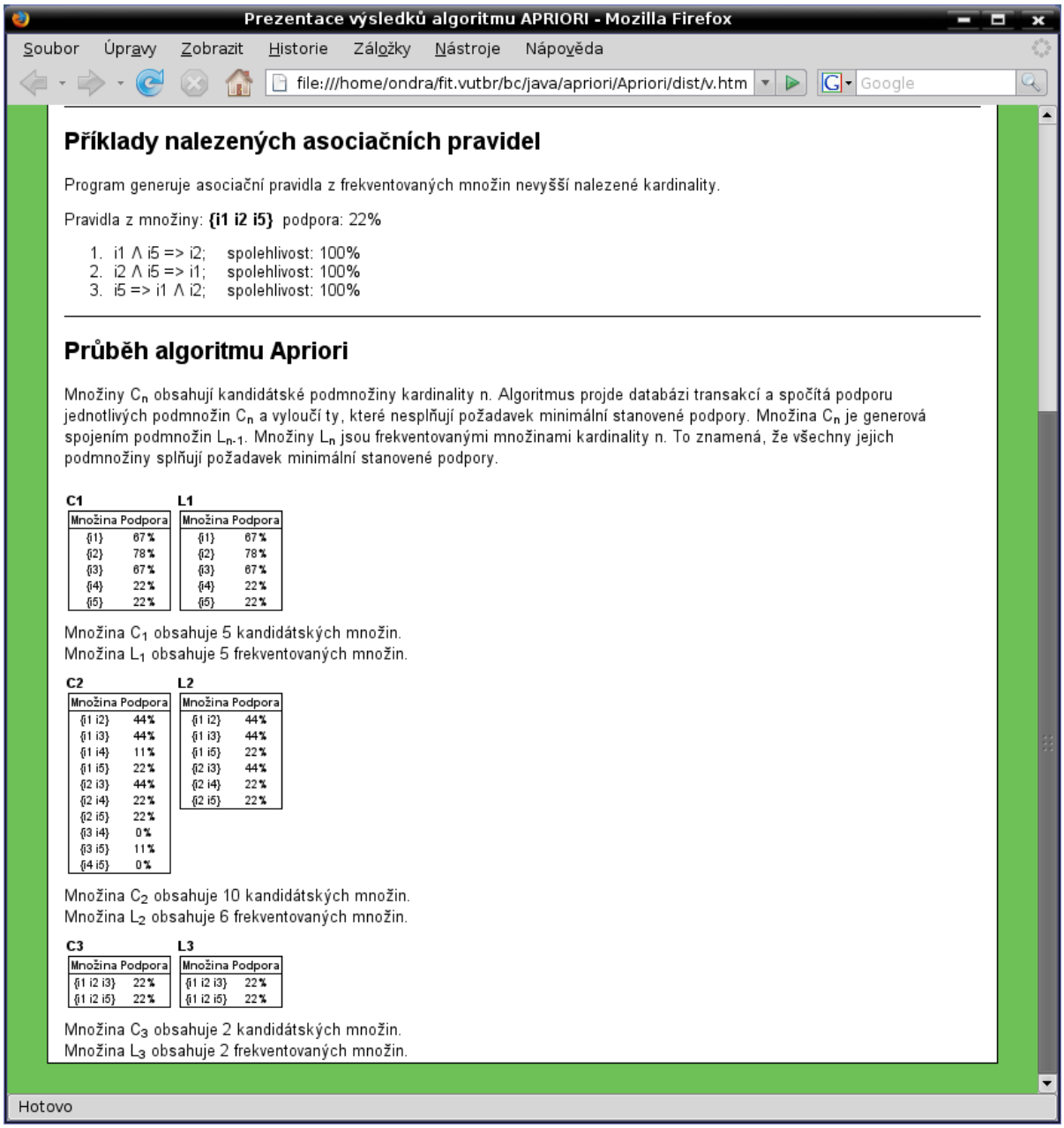
Příloha 1. Ukázka výstupu programu Apriori

Příloha 2. Manuál k programu Apriori

Příloha 3. CD obsahující zdrojové texty programu, zdrojový text bakalářské práce a programovou dokumentaci.

Příloha 1

Ukázka výstupu programu Apriori



Příloha 2

Manuál k programu Apriori

Na přiloženém CD se v adresáři program nachází spustitelný soubor Apriori.jar. Jedná se o konzolovou aplikaci, kterou je možné jak v MS Windows, tak v OS Linux spustit pomocí příkazu *java -jar Apriori.jar*.

Průběh programu:

```
Přejete si načíst stávající soubor transakcí?
(pokud ne, spustí se generátor nového souboru) [A/N]: A


Zadejte cestu k souboru transakcí: databaze.tdb
Zadejte podporu [v %]: 22
Zadejte spolehlivost [v %]: 70
------------APRIORI START------------
Vstupní informace: p=22%, s=70%, položek: 5, transakcí: 9
  Generování kandidátní množiny C1
    - výpočet podpory (průchod databází)
    - uložení frekventované množiny L1
    ...
------------APRIORI KONEC------------
Čas výpočtu: 0 sec

-----Generuji asociační pravidla-----
Čas výpočtu: 0 sec

Zadejte cestu a název pro HTML souboru s výsledky: vysledky.html
```

Nejprve si může uživatel zvolit, zda chce načíst existující databázový soubor nebo chce načíst nový. Pokud se rozhodne pro nový, tak je třeba nejprve zadat cestu a název souboru a pak se program dotáže na počet položek a počet transakcí. Poté je uživatel dotázán na názvy jednotlivých položek (je možné názvy nezdávat). Jakmile existuje databázový soubor, tak se program zeptá na hodnotu podpory a spolehlivosti a pak se spustí algoritmus Apriori. Program vypisuje pouze informace o aktuálně generované množině. Nakonec vypíše dobu výpočtu a vygeneruje asociační pravidla. Poslední informaci, kterou musí uživatel zadat, je název a cesta k HTML souboru s výsledky. Pak je tento soubor programem vygenerován a je možné si jej prohlédnout v libovolném webovém prohlížeči.