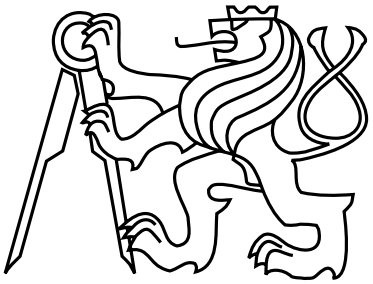
České vysoké učení technické v Praze
Fakulta dopravní
Ústav dopravní telematiky



Diplomová práce

Dolování znalostí z dat v oblasti silniční nehodovosti

Bc. lic. Krzysztof Paweł Urbaniec

Vedoucí práce: doc. Ing. Pavel Hrubeš, Ph.D.

Studijní program: Technika a technologie v dopravě a spojích

Obor: Inženýrská informatika v dopravě a spojích

29. května 2015

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta dopravní
d ě k a n
Konviktská 20, 110 00 Praha 1

K620..... Ústav dopravní telematiky

ZADÁNÍ DIPLOMOVÉ PRÁCE
(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení studenta (včetně titulů):

Bc. Krzysztof Paweł Urbaniec

Kód studijního programu a studijní obor studenta:

N 3710 – ID – Inženýrská informatika v dopravě a spojích

Název tématu (česky): **Dolování znalostí z dat v oblasti silniční nehodovosti**

Název tématu (anglicky): Knowledge Discovery from Road Traffic Accident Data

Zásady pro vypracování

Při zpracování diplomové práce se řiďte osnovou uvedenou v následujících bodech:

- Zpracujte studii využívání aplikací a algoritmů dobývání znalostí z dat v oblasti silniční nehodovosti.
- Zpracujte data Policie ČR a prezentujte identifikované závislosti.
- Definujte doporučující závěry vzhledem k výsledkům provedené analýzy.

Rozsah grafických prací: dle pokynu vedoucího práce

Rozsah průvodní zprávy: minimálně 55 stran textu (včetně obrázků, grafů a tabulek, které jsou součástí průvodní zprávy)

Seznam odborné literatury:

Šimůnek, M. LISp-Miner Šestnáct let vývoje akademického systému pro dobývání znalostí z databází, VŠE, habilitační práce, Praha, 2011

U. M. Fayyad et all, Advances in Knowledge Discovery and Data Mining, The MIT Press, 1996

Hrubeš, P. Analýza statistických dat silniční nehodovosti, ČVUT, habilitační práce, Praha, 2010

Vedoucí diplomové práce: **doc. Ing. Pavel Hrubeš, Ph.D.**

Datum zadání diplomové práce: **25. června 2014**
(datum prvního zadání této práce, které musí být nejpozději 10 měsíců před datem prvního předpokládaného odevzdání této práce vyplývajícího ze standardní doby studia)

Datum odevzdání diplomové práce: **31. května 2015**
a) datum prvního předpokládaného odevzdání práce vyplývající ze standardní doby studia a z doporučeného časového plánu studia
b) v případě odkladu odevzdání práce následující datum odevzdání práce vyplývající z doporučeného časového plánu studia

L. S.

doc. Ing. Pavel Hrubeš, Ph.D.
vedoucí
Ústavu dopravní telematiky

prof. Dr. Ing. Miroslav Svítek
děkan fakulty

Potvrzuji převzetí zadání diplomové práce.

Bc. Krzysztof Paweł Urbaniec
jméno a podpis studenta

V Praze dne 25. června 2014

Poděkování

Děkuji doc. Ing. Pavlu Hrubešovi, Ph.D., za všechno, co pro mne při tvorbě mé diplomové práce udělal. Jen díky Tobě jsem se mohl úspěšně věnovat tak zajímavému tématu.

Děkuji Miroslavu Vanišovi, mému příteli, za neustálou přítomnost. Kdybys mně nepodal ruku ve chvílích, v nichž jsem neměl sílu, nebyl bych dnes tady. Vděčím Ti za mnohem víc, než si dnes myslíš.

Děkuji Ondřeji Hábovi, mému příteli, za cenné jazykové poznámky, díky nimž může být tato práce mnohem lepší. Především však děkuji za všechen čas, jenž jsme spolu strávili na rozhovorech, rozjímáních a poznávání světa. Bez Tebe by byl můj život mnohem plytší.

Děkuji prof. RNDr. Miroslavu Vlčkovi, DrSc., prof. Ing. Zdeňku Votrubovi, CSc., a Dr. Ing. Janu Přikrylovi za čas, jenž mně věnovali v poslední etapě tvorby diplomové práce. Díky Vašim radám a připomínkám jsem mohl tuto práci napsat lépe, než by to bylo možné bez jejich pomoci.

Děkuji celému českému národu za to, že jsem mohl strávit nejdůležitější léta svého mládí obklopen jeho kulturou, zvyky a názory. Obohatil jsi můj život a tím jsi způsobil, že jsem dnes ten, kdo jsem. Díky Tobě více vím, více vidím, více chápu a více jsem – neboť kolik jazyků znáš, tolikrát jsi člověkem.

Dziękuję mojej rodzinie za to, że dała mi szansę iść tam, dokąd idę. Dziś już wiem, że bez Was, kochani, byłbym nikim.
[Děkuji mé rodině za to, že mně dala šanci jít tam, kam jdu. Dnes už vím, že bez Vás, milovaní, bych byl nikdo.]

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Nemám závažný důvod proti užívání tohoto školního díla ve smyslu §60 Zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne 29. 5. 2015 ..... Krzysztof Urbaniec .....

Abstrakt

Tato práce se zabývá otázkou dolování znalostí z databází (DZD) v oblasti silniční nehodovosti. Hlavním cílem je posoudit možnosti aplikace metod data miningu na databázi nehod ve Středočeském kraji a prezentovat dosažené výsledky. Druhým cílem je pokusit se o využití geografických informačních systémů (GIS) v rámci dataminingových úloh včetně zhodnocení výsledků tohoto využití.

Práce se dělí do třech částí. V první části je za účelem uvedení čtenáře do problematiky DZD popsána metodika CRISP-DM, česká metoda GUHA a na ní založený systém LISp-Miner. Důraz je kladen zejména na vztahy, které lze v datech hledat pomocí jeho jednotlivých modulů. Druhá část se věnuje přípravě dat o nehodách před zpracováním s využitím systému LISp-Miner a pravidlům, kterými je vhodné se řídit během práce s tímto systémem. Jsou v ní podrobně popsány všechny úpravy, jaké byly během této fáze na datech provedeny. Třetí část je věnována popisu samotného zpracování dat systémem LISp-Miner. Nachází se zde podrobný popis osmi typů dataminingových úloh realizovaných na databázi nehod včetně úloh využívajících GIS. Důraz je kladen na nastavení každé úlohy a interpretaci výsledných hypotéz. V závěru jsou výsledky zhodnoceny a na jejich základě jsou formulována doporučení pro další výzkum.

Klíčová slova: data mining, dolování znalostí z dat, DZD, geografické informační systémy, GIS, LISp-Miner, metoda GUHA, silniční nehody.

Abstract

The thesis considers the process of knowledge discovery in databases of road traffic accidents (KDD). The main purposes of the thesis is to evaluate possibilities of applying data mining methods to the database of traffic accidents in Central Bohemia and to present the achieved results. The second purpose is to attempt to use the geographic information systems (GIS) in data mining tasks and to evaluate the results.

The thesis is divided into three parts. In the first part, the CRISP-DM process model, the Czech GUHA method, and the LISp-Miner system based on this method are depicted in order to introduce the reader to the topic of KDD. The focus lays especially on the relations to be searched in the database using particular modules of the LISp-Miner system. The second part describes the phase of data preparation before working with the LISp-Miner system and the rules of using it. All modifications and adjustments carried out during this phase are described here. The third part considers the process of data analysis using the LISp-Miner system. The detailed description of the eight types of executed data mining tasks including the tasks incorporating GIS information is to be found here. Particular attention is paid to the settings of each task and the interpretation of the achieved results. In the conclusion, the results are evaluated and possible directions to continue the research are suggested.

Keywords: data mining, geographic information systems, GIS, GUHA method, knowledge discovery in databases, KDD, LISp-Miner, road traffic accidents.

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Pro informační civilizaci, za jakou se dnes považujeme, jsou data jedním ze základních prvků světa. S rostoucím výkonem výpočetní techniky rostou jejich objemy: ukládáme data o počasí, o telefonních hovorech, o příjmech a výdajích, o vzdělání, zkrátka o všem, co nám jen přijde na mysl. Většinou jsou tyto údaje ukládány za účelem získání z nich konkrétních informací a potažmo znalostí. Konkrétní úkony, které se dále s daty provádí, však obvykle zdaleka nevyčerpávají obrovský informační potenciál, který tato data skrývají. Z toho důvodu vznikl docela nedávno přístup ke zpracování dat nazývaný explorační analýza. Jednou z jejích metod je dolování znalostí z databází (neboli data mining) a právě jím se tato práce zabývá.

Doprava je inženýrskou oblastí, ve které se mimořádně obsáhlé soubory dat doslova „povalují“ všude kolem nás. Máme k dispozici miliony záznamů z mýtných bran a dopravních detektorů. Tato data jsou sbírána za konkrétními účely. Situace v oblasti sběru dat o dopravních nehodách je jiná – tato data jsou ukládána příslušnými složkami především za účelem evidování nehod a následných správních řízení. Ve světě bylo dosud realizováno mnoho výzkumných prací, které využívaly tohoto typu dat při dataminingovém zpracování, což je důkazem, že je to zajímavé téma, kterému stojí za to se věnovat.

První výsledky tohoto typu výzkumů dopravních nehod pocházejí z roku 1995. Typicky dataminingový výzkum proběhl např. na začátku 21. století ve Velké Británii na objednávku hrabství Hampshire – byl to výzkum dopravní nehodovosti na jeho území ([1], viz také 1.3.3). V jeho rámci byly nalezeny mj. závislosti mezi číslem silnice a příčinou nehody. Výsledky byly předány státním institucím a shledány jako zajímavé. Krátce potom následoval výzkum nehod v regionu West Midlands (popis výzkumu se nachází v [2]), jehož součástí bylo opět hledání tohoto typu pravidel. Tentokrát byly hlavními zkoumanými parametry značka vozidla, věk obětí a závažnost nehody.

Po několika letech se toto téma stalo předmětem tolika výzkumných prací, že začaly vznikat i studie, jejichž náplní byl přehled metod používaných při dobývání znalostí z dat týkajících se nehod. Příkladem je indická studie [3] z roku 2009, která popisuje 17 (sic!) různých přístupů a jmenuje 18 (sic!) různých nástrojů použitých ve zpracování dat v oblasti nehodovosti. Popisované jsou výzkumy prováděné v mnoha různých zemích světa, např. v USA, Jižní Koreji či Velké Británii. Od té doby jsou neustále zkoumány další možnosti a prováděny další práce. Např. v roce 2011 byla publikována studie zkoumající perspektivy aplikace mj. rozhodovacích stromů či jiných moderních nástrojů zpracování dat na databáze nehod [4]. Metody data miningu začínají být také používány při výzkumu věnujícím se problematice Smart Cities a udržitelné mobilitě [5].

Také u nás najdeme publikace věnující se této problematice. Příkladem může být studie zabývající se dobýváním znalostí z dat o nehodách v Etiopii, která byla publikována v roce

2012 [6][1]. Spolupodíleli se na ní autoři z Vysoké školy báňské – Technické univerzity Ostrava. Publikace na toto téma jsou však v České republice ojedinělé.

V souvislosti s čím dál rychlejším vývojem aplikací data miningu v otázce dopravních nehod si v rámci diplomové práce kladu tři cíle. Prvním z nich je navázání na výše zmíněné výzkumné práce a realizace úloh dolování znalostí z databází na datech týkajících se silničních nehod ve Středočeském kraji. Dle mých znalostí se přes bohaté světové zkušenosti ještě nikdo v České republice realizaci takového úkolu v oblasti nehodovosti nevěnoval, je to tedy šance získat dosud neobjevené poznatky a zahájit v českém prostředí nový výzkum, ve kterém lze po ukončení mého studia pokračovat. Za tímto účelem budu pracovat s daty získanými od Policie České republiky, která jsou tvořena záznamy o přibližně 800 000 nehodách, které se v ČR staly v letech 2007–2013.

Druhým cílem je pokus o využití geografických informačních systémů (GIS) v procesu dobývání znalostí z databází. Dle mých znalostí se o takové propojení ještě nikdy dosud nikdo nepokoušel, jedná se tedy o aplikaci zcela nového přístupu ke GIS a data miningu. Pokud budou výsledky navržených postupů slibné, může pokračování v tomto směru vést k objevům dosud neznámých zákonitostí a v budoucnu k využití nově nabytých znalostí pro zlepšení dopravní infrastruktury. V rámci realizace tohoto úkolu použiji GIS k vytvoření nových, geografických údajů o nehodách a pokusím se tak nalézt nové hypotézy v této oblasti.

Posledním, avšak podle mého názoru nejdůležitějším cílem, jaký si zde kladu, je vytvoření kompletní příručky, která umožní zcela neznalému čtenáři zahájit svůj vlastní výzkum založený na dobývání znalostí z databází. Tato příručka by měla čtenáři především vysvětlit, čím data mining je, v jakých principech spočívá a jakými metodami se provádí. Dále by měla prezentovat softwarové prostředí, ve kterém lze výzkum provádět, a napovědět, jakým způsobem lze takové prostředí vytvořit na vlastním počítači. Měla by také čtenáři pomoct při přípravě dat a upozornit ho na nejdůležitější kroky, které je v této etapě nutno podniknout. Konečně by také měla názorně ukázat, jak se samotné dobývání znalostí z dat provádí, jakých výsledků lze dosáhnout, jak je lze prezentovat a jakým způsobem interpretovat. Řečeno jedním slovem, měla by být pro čtenáře **inspirací** pro budoucí práci.

Práce je rozdělena do pěti kapitol. První z nich se věnuje uvedení čtenáře do problematiky dolování znalostí z databází. Tato kapitola seznámí čtenáře s dnešním přístupem k této otázce, s původní českou metodou GUHA a s na jejím základě vytvořeným dataminingovým systémem LISp-Miner. Druhá kapitola rozšiřuje obecný obraz o podrobnější popis základních procedur, které jsou v systému LISp-Miner implementovány, a to včetně popisu závislostí, které jsou těmito procedury vyhledávány. Třetí kapitola se věnuje fázi předzpracování dat: nachází se v ní popis použitého prostředí, popis samotných dat, metody jejich čištění a způsoby přípravy pro vlastní zpracování. Jejím pokračováním je čtvrtá kapitola, ve které uvádím důležitá pravidla a doporučení, kterými je vhodné se řídit během práce se systémem LISp-Miner. Je to zároveň úvod k poslední, páté kapitole, ve které se nachází popis několika vybraných dataminingových úloh realizovaných na databázi silničních nehod. Tento popis obsahuje způsob definování úlohy, popis získaných hypotéz a jejich interpretaci. V závěru jsou hodnoceny výsledky celé práce a na jejich základě jsou stanoveny návrhy dalších směrů výzkumu, který může být pokračováním toho, co bylo touto prací zahájeno.

[1] Zde je záhodno zmínit, že časopis *Neural Network World*, ve kterém byla tato práce publikována, vychází ve spolupráci s ČVUT v Praze Fakultou dopravní. Je to pro studenta této fakulty zvláště silná motivace pro realizaci výzkumu, který by se zabýval nehodovostí v České republice, přímo na své alma mater.

Kapitola 1

Stručný úvod do problematiky dolování znalostí z databází

Ve většině případů se inženýrská práce vyznačuje předem daným, jasně definovaným cílem. Tím může být zjišťování platnosti konkrétní hypotézy, nalezení řešení daného problému, optimalizace už existujícího řešení nebo jakýkoliv jiný, snadno představitelný úkol. Za tímto účelem se pak většinou provádí výzkum, s čímž je mnohdy spojen sběr dat, která mají potvrdit nebo zavrhnout posuzované předpoklady. Je to tzv. konfirmační analýza. Existuje však také přístup zcela odlišný od běžného inženýrského působení. Princip tohoto přístupu spočívá v obrácení výše popsané posloupnosti – v dříve získaných datech se hledají ještě neznámé závislosti či zákonitosti, na jejichž základě se pak formulují nové vědecké poznatky. Tento proces se nazývá explorační analýza.

Snad nejznámějším způsobem zpracování dat spadajícím do kategorie explorační analýzy je dobývání znalostí z databází neboli data mining. Jedná se o automatické vyhledávání závislostí ve velkých souborech dat. Od obecně pojaté explorační analýzy dat se data mining liší oblastí zájmu: zatímco se explorační analýza soustředí především na uchopení a porozumění fenoménům reprezentovaným konkrétními daty, data mining se zaměřuje primárně na hledání závislostí a z toho plynoucí řešení, aniž by k tomu bylo nutné jejich jasné pochopení a stanovení jejich příčin.

Účelem této kapitoly je seznámení čtenáře s problematikou dobývání znalostí z databází v rozsahu, jaký je nezbytný pro porozumění obsahu této práce. V první části je stručně uveden proces data miningu bez vztahu ke konkrétním aplikacím či metodám. Následně je popsána původní česká metoda GUHA, na jejímž základě byla v rámci této práce provedena nejpodstatnější část analýzy dat. V koncové části kapitoly se nachází popis systému LISp-Miner, který je nejpokročilejší aplikací metody GUHA a který byl hlavním nástrojem, který jsem používal.

Čtenář obeznámený s danou tematikou může přeskočit jednotlivé sekce této kapitoly, aniž by to způsobilo potíže při čtení dalších částí práce. Naopak pokud čtenář projeví zájem o podrobnější výklad týkající se metody GUHA a systému LISp-Miner, najde ho v pracích zmíněných v sekci „Literatura“ (především [7] a [8]).

1.1 Obecně o dolování znalostí z databází

Dobývání znalostí z databází (ang. *Knowledge Discovery in Databases*) či zkráceně „data mining“ (z ang. doslova „dolování z dat“, „vytěžování dat“) je obecně procesem chápaným jako získávání netriviálních informací ze souborů dat. Přesná definice není stanovena z důvodu existence velkého počtu metod a postupů, kterými lze toto získávání realizovat. Nejlepší definici jsem našel v sekci 2.1 v [7]:

> Dolování znalostí z databází je proces analýzy velkého množství dat prostřednictvím netriviálních technik pro vyhledávání opakujících se vzorů, pravidelností nebo naopak nepravidelností, které jsou potenciálně využitelné.

Z této obecné definice neplyne, jakým způsobem se takové znalosti hledají. Za účelem alespoň rámcového sjednocení postupů a standardizace bylo navrženo několik metodik, které mají za účel zaručit dosažení kvalitních výsledků i nezkušenými badateli. Díky tomu krátký popis jedné z nich umožní čtenáři lepší porozumění, v čem spočívá princip data miningu. Autoři systému LISp-Miner, se kterým jsem pracoval, navrhli svoje dílo podle metodiky CRISP-DM. Z toho důvodu se krátký popis právě této metodiky pro seznámení čtenáře s procesem dolování z dat jeví jako nejlepší volba.

Metodika CRISP-DM

Název CRISP-DM je odvozen od anglického názvu *Cross Industry Standard Process for Data Mining*. Podle této metodiky je proces data miningu rozdělen do šesti základních fází. Tyto fáze jsou definovány autory metodiky takto (viz [9]):

1. **Porozumění problematice** – je to základní a nejdůležitější fáze, pokud výsledkem má být reálně aplikovatelná znalost. Jejím obsahem je pochopení cílů daného projektu a na jejich základě formulování dataminingové úlohy včetně předběžného plánu dosažení výsledků.

2. **Porozumění datům** – v této fázi probíhá seznámení s daty, vyhodnocení jejich využitelnosti i kvality a hledání zajímavých podmnožin, na kterých se budou formulovat hypotézy. Patří sem i vlastní (počáteční) sběr dat.

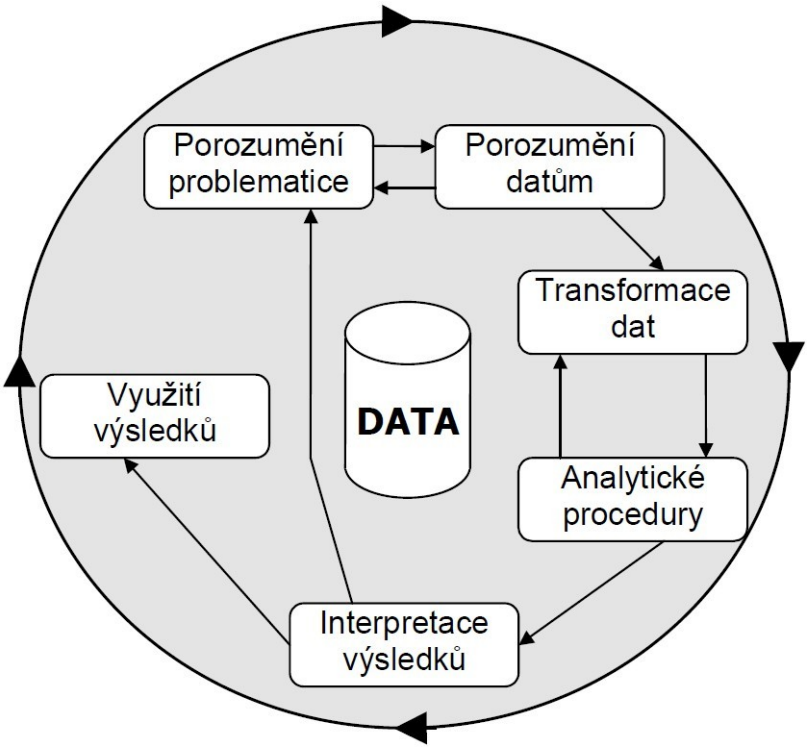
3. **Transformace dat** – tato fáze spočívá v přípravě dat pro konkrétní zpracování. Zahrnuje v první řadě čištění a předzpracování dat, ale i generování nových sloupců z existujících dat či úpravy existujících záznamů tak, aby byly vhodné k aplikaci analytických procedur.

4. **Analitické procedury** – klíčová fáze, ve které probíhá „vlastní data mining“, tj. zpracování dat příslušnými procedurami a hledání v nich závislostí. Její součástí je volba vhodné metody, její aplikace, definování tvaru hledaných zákonitostí apod.

5. **Interpetace výsledků** – fáze zhodnocení výsledků tvorby modelu. Toto zhodnocení probíhá především v otázce efektivity i kvality a splnění definovaných požadavků. Kromě toho fáze zahrnuje i interpretaci dosažených výsledků neboli odvození a definování nově získaných znalostí.

6. **Využití výsledků** – aplikace výsledků v praxi. Je to samostatná kapitola, bez které celý postup nemá reálný význam, která však není přímo obsahem zpracování v procesu dobývání znalostí z databází.

Obrázek 1.1 znázorňuje posloupnost fází a jejich provázání. Vnější kruh znázorňuje cyklický charakter dobývání znalostí z databází. Tato neustále se opakující posloupnost kroků je nejdůležitější vlastností této metodiky. Díky poznatkům získaným ve fázi interpretace výsledků se zvyšují znalosti dané problematiky. Celý proces lze zopakovat, tentokrát lépe a hlouběji, a „dobýt“ tak více ještě kvalitnějších znalostí.



Obrázek 1.1: Schematické znázornění jednotlivých fází metodiky CRISP-DM. Zdroj: [7]

Za povšimnutí stojí především dva malé cykly, které lze vidět na obr. 1.1. První je dvojice fází „porozumění problematice“ a „porozumění datům“. V průběhu seznamování se s daty získává badatel větší přehled o dané problematice, jelikož jednoduché souvislosti, jaké lze vidět na první pohled, či dokonce samotný charakter dat a v nich zaznamenaných veličin může prohloubit jeho představy a umožnit lepší porozumění celé oblasti. Na základě toho lze například provést dodatečný sběr dat či všimnout si dalších potenciálně zajímavých vztahů v datech.

Druhý cyklus znázorňuje úzké provázání fáze transformace (přípravy) dat s fází analytických procedur (zpracování). Tato souvislost je zřejmá: v následku provedení zpracování a posouzení jeho výsledků badatel často zjistí, že to, co obdržel, neodpovídá jeho představám či nesplňuje požadavky na kvalitu. Málokdy se povede dosáhnout kvalitních výsledků hned napoprvé. Za účelem zlepšení je mnohdy potřeba opakovaně transformovat data a znova na

ně aplikovat zvolené postupy, popř. tyto postupy měnit, aby vyhovovaly upraveným datům. Připomíná to trochu postupné zaměřování dělostřelecké palby – napoprvé se obvykle netrefí, ale postupným zpřesňováním lze dosáhnout cíle.

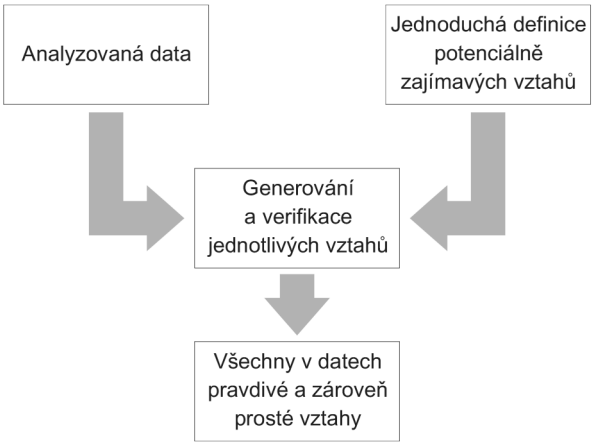
Až na jednu jsou všechny fáze metodiky CRISP-DM jednoduše představitelné, pokud se jedná o jejich obsah. Výjimku tvoří fáze analytických procedur. V této fázi probíhá aplikace dataminingových procedur a metod. Způsob automatického analytického zpracování dat je hlavním tématem výzkumu v oblasti dobývání znalostí z databází. Pro přiblížení čtenáři možného způsobu provedení takového zpracování je v následující sekci umístěn stručný popis jedné z existujících metod – metody GUHA.

1.2 Metoda GUHA

Poznámka: *Níže uvedený popis metody GUHA je částečně převzat z autorovy bakalářské práce ([10]).*

GUHA (ang. *General Unary Hypotheses Automaton*) je „metoda automatického hledání generálních unárních hypotéz“ pravdivých v daném (empiricky získaném) souboru dat. Poprvé byla tato metoda prezentována v listopadu 1965 na Druhé konferenci o kybernetice v Praze (zápis z přednášky byl publikován v [11]). Její autoři ji na začátku používali ve fyziologickém výzkumu, podotkli však, že podle nich má tato metoda předpoklady pro použití v libovolné oblasti, což bylo během několika desítek let potvrzeno jejím širokým použitím a dalším vývojem metody jako takové.

Jak je zřejmé už ze samotného názvu, GUHA je automatem sloužícím ke generování hypotetických závislosti plynoucích ze vstupních dat. V případě běžné práce se pro zjišťování platnosti daného vztahu určuje výběrový soubor a aplikují statistické metody. Musí se dodržet přesná pravidla a vlastnosti souboru i celé populace, aby odvozené poznatky byly ve skutečnosti platné, přesto však platí jenom s určitou pravděpodobností. Automatizace tohoto procesu a zároveň generování hypotéz umožňuje prozkoumání skutečně všech vztahů,



Obrázek 1.2: Konceptuální schéma metody GUHA. Zdroj: [7]

které v daném případě platí (anebo neplatí). Autoři metody ji v [11] sami přirovnávají k výlovu rybníka vypuštěním – teprve po vypuštění rybníka máme naprostou jistotu, že známe právě všechny ryby, které v rybníku jsou, a nemusíme tak odvozovat pouze pravděpodobné důsledky.

Další zajímavou vlastností metody GUHA je, že může sloužit jako „náhrada intuice“, tedy že generování hypotéz a nápadů, „co by mohlo v daném případě platit“, není její jedinou (byť je primární) funkcí. Zřejmé je, že pokud výzkumník nějakou hypotézu předem má, může automaticky prozkoumat její platnost v celém souboru dat. Velmi užitečná je však GUHOU nabízená korekce hypotézy, které jsou sice neplatné nebo špatně formulované, ale platí po jistých úpravách. Může tedy sloužit také k „vylepšování“ hypotéz.

1.2.1 Definice a pojmy

Závislosti generované GUHOU jsou závislosti přesně daného typu, rovněž zmíněného v názvu. První návrh obsahoval zavedení pomocí formálních (především matematických) definic pojmu těchto „generálních unárních hypotéz“. Toto zavedení je nutno velmi stručně a zjednodušeně přiblížit pro ujasnění, jaká je oblast aplikace metody GUHA, a pro pochopení, jaké hypotézy je schopna generovat. Níže uvedené definice jsou převzaty z [11] a upraveny do zkrácené podoby.

> Nechť je dán model $\mathcal{M} = \langle M, P_1, P_2, \ldots, P_n \rangle$. M je v tom případě množina objektů a $P_1, P_2, \ldots, P_n$ jsou vlastnosti. Jestliže a bude objekt z množiny M a P_i bude i-tou vlastností, pak výrok $P_i(a)$ čteme „a má i-tou vlastnost“ a nazýváme *unárním predikátem* na modelu $\mathcal{M}$. Lze zavést n takových unárních predikátů $P_i(x)$. Dále lze z těchto predikátů, jež jsou zároveň elementárními výrokovými formulemi, vytvořit pomocí logických spojek další výrokové formule; označíme takovou formuli Φ. Připojíme zpředu k Φ generální kvantifikátor $(\forall x)$. Takto sestrojenou formuli nazýváme *generální unární hypotézou* (GUH). Pokud je generální unární hypotéza sestrojená k formuli Φ pravdivá v modelu $\mathcal{M}$, znamená to, že pro všechny objekty modelu je mezi vlastnostmi $P_i(x)$ souvislost Φ.

Z výše uvedeného plyne, že generální unární hypotéza je logický výrok formulovaný na základě vybraných vlastností, posuzovaný z hlediska platnosti na celé množině objektů, přičemž všechny vlastnosti a množina objektů tvoří zkoumaný model; je to zkrátka vyjádření jisté skutečnosti týkající se daného objektu. Tato hypotéza nemusí obsahovat všechny vlastnosti z modelu, nesmí však obsahovat vlastnosti, které do modelu nepatří. Generální unární hypotézy jsou např. výroky *Řidiči jezdící červenými škodovkami jsou vyšší než 180 cm*, *Rychlost aut starších než 10 let na tomto úseku dálnice nepřekračuje 120 km/h* či *Každý řidič, který má řidičský průkaz méně než tři roky, obdržel za rok alespoň dvě pokuty.*

Je třeba ujasnit několik dalších pojmů používaných v této práci. Potřebné definice lze ve stručné a srozumitelné formě najít v [12], nachází se v nich však formální chyby, které jsem upravil. Podobu po úpravě a vynechání málo důležitých z hlediska této práce pasáží uvádím níže:

- *predikát* – symbolické jméno veličiny,
- *formule* – predikát (v tom případě je to elementární formule) nebo více predikátů složených pomocí logických spojek negace, konjunkce a disjunkce,
- *kvantifikátor* – symbolické jméno zobrazení, které určuje kvantitativní intenzitu souvislosti; představuje „druh“ zjištěného vztahu v datech,
- *formální sentence* – zápis tvaru
$$f_1 \quad q \quad f_2,$$
kde f_1, f_2 jsou formule a q je kvantifikátor, jehož pravdivost v datech se testuje,
- *pravdivá sentence* – sentence, pro kterou funkce kvantifikátoru vrátila hodnotu 1; sentence pravdivá v datech,
- *antecedent* – formule vyskytující se uvnitř sentence na levé straně kvantifikátoru,
- *sukcedent* – formule vyskytující se uvnitř sentence na pravé straně kvantifikátoru,
- *cedent* – sukcedent, antecedent nebo podmínka.

1.2.2 Hlavní princip funkce

Jak už bylo zmíněno, GUHA generuje na daných množinách vstupů všechny možné hypotézy a testuje, zda jsou ve vstupních datech podporovány. Testování se neprovádí na základě hodnot veličin popisujících jednotlivé objekty, nýbrž na základě tzv. *frekvencí*. Pro přiblíženi tohoto pojmu použiji opět postup uvedený v [12]. Cílem je přiblížit čtenáři princip, na jakém GUHA funguje, nikoliv obecné formální definování, pro jednoduchost se tedy mohu omezit na binární veličiny.

> Nechť je M tabulka vzniklá pozorováním n dvouhodnotových veličin $X_1, \ldots, X_n$. Pro každou n-tici možných hodnot veličin, tj. $e = \langle e_1, \ldots, e_n \rangle \in \{0,1\}^n$, definujeme *frekvenci* $fr(e, M)$ jako počet objektů z M, pro které jsme pozorovali hodnoty veličin rovné $e = \langle e_1, \ldots, e_n \rangle$. Konkrétně pro $n = 2$, tj. tabulku se dvěma řádky a sloupci X_1 a X_2, definujeme čtyři frekvence a, b, c, d. Frekvence jsou počty objektů z níže uvedené tabulky, tzv. *frekvenční (čtyřpolní) tabulky*, pro kterou platí, že obě veličiny (v řádku a ve sloupci) mají zároveň hodnotu 1:
>
> | | X_2 | $\neg X_2$ | |
> |---|---|---|---|
> | X_1 | a | b | r |
> | $\neg X_1$ | c | d | s |
> | | k | l | m |
>
> k, l, r, s jsou sumy hodnot v tabulce po řádcích nebo sloupcích, m je suma všech hodnot v tabulce. V obecném případě má tabulka n řádků a sloupců tvořených na stejném principu.

Nad frekvencemi se definuje řada kvantifikátorů. Ty se v závislosti na prezentovaném pravidle dělí na asociační, implikační nebo korelační. Kvantifikátor se definuje jako funkce frekvencí; pokud je výsledkem 1, zkoumané je pravidlo přijato.

Asociační kvantifikátory se označují $\sim$ ($A \sim B$ čteme „A (asi, většinou) souvisí s B“). Asociační kvantifikátor říká, že shody v daném případě převažují nad neshodami. Jsou to např.:

- prosté vychýlení,
- Fischerův kvantifikátor,
- χ^2 kvantifikátor.

Implikační kvantifikátory se označují $\Rightarrow$ ($A \Rightarrow B$ čteme „A (asi, většinou) je příčinou B“). Mohou to být:

- fundovaná implikace – implikace se splněním minimální podpory a spolehlivosti,
- dolní kritická implikace,
- horní kritická implikace.

Dvě poslední implikace jsou založeny na statistických testech, přičemž první z nich indikuje přijetí hypotézy, že je podmíněná pravděpodobnost sukcedentu za podmínky antecedentu větší než zadaná hodnota, druhá pak indikuje nezamítnutí hypotézy, že je tato pravděpodobnost větší nebo rovna zadané hodnotě. Díky tomuto rozdílu můžeme zvolit kvantifikátor v závislosti na tom, kterou ze statistických chyb chceme omezit.

Korelační kvantifikátory se označují corr (A corr B / F čteme „za podmínky F hodnoty A a B (asi, většinou) korelují“). Všechny jsou založeny na pojmu *pořadí*. Tento pojem předpokládá využití dat vzniklých pozorováním dvou reálně-hodnotových veličin a využití počtů objektů, pro které hodnota jedné z veličin je menší než hodnota téže veličiny pro daný objekt. Používají se mj. tyto tři druhy:

- Spearmanův kvantifikátor,
- Kendallův kvantifikátor,
- pořadově ekvivalenční kvantifikátor.

V další části práce se také nachází kvantifikátory používané v jednotlivých GUHA procedurách implementovaných v systému LISp-Miner (viz tab. 2.1, 2.2, 2.3 a 2.4).

1.2.3 GUHA procedury

Metodu GUHA tvoří řada procedur zabývajících se hledáním určitého typu závislostí, tedy procedur založených na různých typech kvantifikátorů. Jedněmi z původních procedur, které výrazným způsobem ovlivnily další vývoj metody, jsou procedury ASSOC, IMPL a CORREL, které hledají příslušně sentence s asociačními kvantifikátory, implikace a vysoké podmíněné korelace. Velmi dobrý a srozumitelný popis těchto procedur včetně jejich

předpokladů je uveden v [12], zde uvedu jenom zkrácenou verzi jako nástin způsobu práce metody GUHA.

Procedura ASSOC hledá asociace mezi formulemi, které jsou ve tvaru elementární konjunkce. Tvar výsledných sentencí se zadává tak, že se určuje asociační kvantifikátor, jeho parametry a povolené tvary antecedentu a sukcedentu. Ty poslední se určují tak, že se zadají čtyři množiny predikátů: důležité predikáty a ostatní vyšetřované predikáty, a to zvlášť pro antecedenty a sukcedenty. Generují se jen sentence, které obsahují aspoň jeden důležitý predikát v antecedentu a sukcedentu.

Algoritmus procedury probíhá zhruba tak, že se první generují sentence s jedním predikátem v antecedentu i jedním v sukcedentu a postupně se testuje pravdivost všech přípustných kombinací. Dále se prodlužuje antecedent nebo sukcedent a opět se testuje pravdivost všech přípustných kombinací. Proces se opakuje až do dosažení maximálních stanovených velikostí.

Procedura IMPL generuje pravdivé sentence s implikačními kvantifikátory. Rozdíl oproti proceduře ASSOC spočívá v tom, že sukcedenty jsou ve tvaru elementární disjunkce místo konjunkce. Jinak jsou vstupy stejné a algoritmus výpočtu velmi podobný.

Procedura CORREL generuje elementární konjunkce, pro které je podmíněná pravděpodobnost dvou vybraných reálně-hodnotových veličin v datech vysoká, tedy vydává sentence tvaru

$$p_1 \text{ corr } p_2 \ / \ f,$$

kde p_1 a p_2 jsou zvolené veličiny, corr je korelační kvantifikátor a f je podmínka ve tvaru elementární konjunkce. Během práce procedury jsou p_1, p_2 a corr pevné, mění se jen f. Vstupy procedury jsou veličiny p_1 a p_2, které zkoumáme, užitý kvantifikátor, povolený tvar podmínky tvořený dvěma množinami (množinou důležitých predikátů a množinou ostatních vyšetřovaných predikátů) a maximální povolená délka podmínky. Generují se pouze podmínky, které obsahují alespoň jeden důležitý predikát.

Algoritmus hledání pravdivých sentencí je založen na postupném generování podmínek povoleného tvaru, jejich postupném prodlužování až do maximální délky a testování pravdivosti vytvořených sentencí.

1.2.4 Aplikace

Metoda GUHA je používána v mnoha počítačových programech a systémech už od doby svého vzniku. Souhrn všech aplikací je zhotoven v [7] a tento souhrn ve zkrácené formě obsahující přínosy jednotlivých aplikací zde uvádím. Autory jednotlivých aplikací, platformu, odkazy na popisující je literaturu a podrobnější popisy mohou zájemci najít ve zmíněné práci.

- GUHA – 1965

  První implementace; byla vytvořena v roce 1965 současně se stanovením samotné metody GUHA; pracovala pouze s binárními daty a generovala disjunkce predikátů.

- GUHA-S – 1968

  „Statistická“ implementace obohacená o práci s konjunkcemi predikátů a Fisherův statistický test.

- Trojhodnotová GUHA – 1971

  Implementace konceptu chybějící informace pomocí trojhodnotové logiky (ano-ne-?).

- Asociační a implikační GUHA – 1977

  Implementace zobecněných kvantifikátorů.

- GCL GUHA – 1985

  Zavedené hypotézy s podmínkou; optimalizace.

- PC-GUHA – začátek 90. let 20. Století

  Převod na platformu IBM PC; přidání vstupních, výstupních a interpretačních obrazovek.

- 4ft-Miner – 1997

  Implementace procedury DB-ASSOC (podrobněji viz 1.3.1).

- GUHA +/- – 1998

  Zavedení možnosti automatické kategorizace atributů; další kvantifikátory.

- LISp-Miner – 1999

  Implementace celkem osmi GUHA procedur a několika dodatečných procedur i modulů.

- Ferda – 2005

  Obsahuje sedm z osmi GUHA procedur implementovaných v LISp-Mineru; studentský projekt na Vysoké škole ekonomické v Praze. Vývoj přerušen v roce 2010.

1.3 Systém LISp-Miner

LISp-Miner (LISp – *Laboratoř inteligentních systémů Praha*) je akademický systém vyvíjený pedagogy a studenty VŠE v Praze. Jedním z jeho základních úkolů je podpora výzkumných aktivit v oblasti data miningu, především studentských. Systém je postupně vyvíjen od roku 1996, hlavními autory jsou Jan Rauch a Milan Šimůnek.

Systém je volně přístupný pro všechny zájemce. Na domovské stránce projektu LISp-Miner ([13]) se v sekci „Download“ nachází instalační soubory, navíc lze na stránkách najít prezentace demonstrující použití systému, tutoriály, vzorový soubor dat aj. V sekci „KDD procedures“ se nachází detailní výpis všech procedur, jež LISp-Miner používá. Zájemci mohou v sekcích „Research“ a „Applications“ najít také podrobné popisy nebo odkazy na popis aplikací ve výzkumu a výuce, dále pak odkazy na literaturu použitou při tvorbě systému LISp-Miner a také na literaturu vytvořenou na základě jeho použití.

Určení systému pro využití studenty ve výzkumných aktivitách, volný přístup a velmi bohatý popis použití byly důvody, proč jsem se rozhodl použít v rámci této práce právě systém LISp-Miner. Tuto podkapitolu věnuji velmi stručnému popisu systému: historii jeho vzniku, současné podobě a vybraným aplikacím.

Během doby, která uplynula od obhajoby mé bakalářské práce, prošel systém LISp-Miner dalším vývojem a změnila se jeho architektura. V souvislosti s tím nedoporučuji čtenáři

čerpat z této práce informace týkající se tohoto systému. Část pasáží této podkapitoly jsem z ní převzal (např. většinu sekce „Historie“), udělal jsem to však s ohledem na provedené změny i vývoj a jenom v místech týkajících se oblastí, ve kterých se tyto změny nekonaly.

Hlavními zdroji informací uvedených v této sekci jsou habilitační práce M. Šimůnka [7], která se věnuje vývoji jeho „dítěte“, kterým je LISp-Miner, od vzniku až do roku 2011, a kniha „Dobývání znalostí z databází, LISp-Miner a GUHA“ [8], která je obsáhlým, podrobným a nejaktuálnějším shrnutím dvaceti let vývoje systému LISp-Miner až do dnešního dne. Veškeré zájemce o podrobnější výklad v otázkách, kterým se moje práce nevěnuje, proto odkazuji na tyto dvě publikace.

1.3.1 Historie

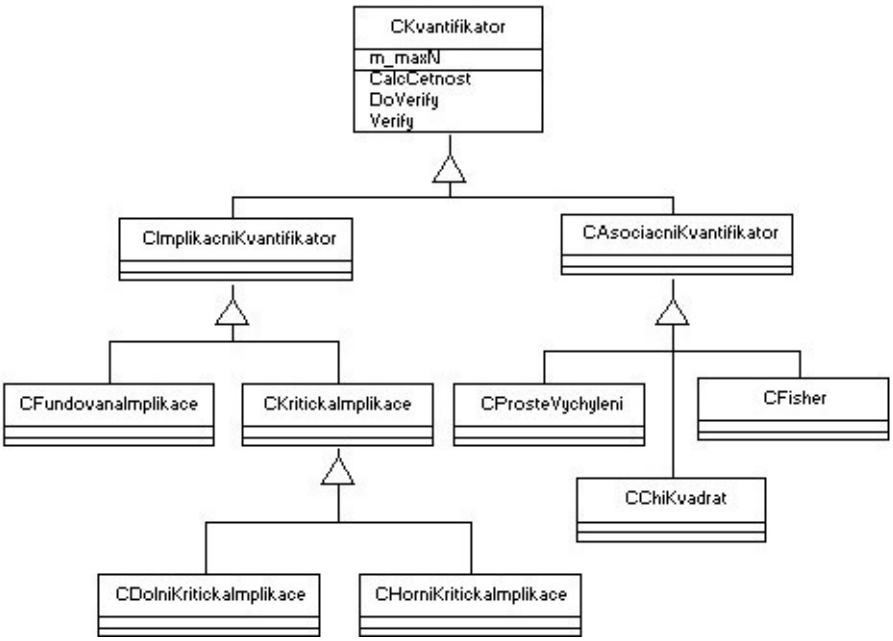
Autoři datují vznik systému LISp-Miner (nebo přesněji jeho původní podoby) na přelom let 1995/96. Byla to nová implementace metody GUHA, vycházející z jejích předchozích implementací. Úplně první částí byla naprogramovaná v roce 1996 dílčí část GUHA procedury DB-ASSOC, konkrétně kvantifikátory a verifikace relevantních otázek. Na obr. 1.3 se nachází objektový model tehdy naprogramovaných kvantifikátorů.

Implementace těchto kvantifikátorů se ukázala být zajímavou a proto začal vývoj ucelené aplikace. Po dvou rocích byl vytvořen 4ft-Miner – monolitická, jednouživatelská aplikace, pokrývající všechny kroky data miningu (2.–5. podle metodiky CRISP-DM, viz 1.1): porozumění datům, předzpracování, vlastní analýzu (v tom případě ve smyslu procedury DB-ASSOC) a následnou interpretaci výsledků. Dodnes je 4ft-Miner jednou s procedur celého systému LISp-Miner, navíc jako samostatná aplikace naznačil vhodný směr pro podporu uživatele v seznámení s daty a jejich předzpracování.

Další vývoj aplikace 4ft-Miner směřoval k zavedení rodin veličin. Časem se ukázalo, že rozvíjení tohoto konceptu, zvláště v monolitické aplikaci, začíná být příliš složité. Používání nových možností začalo být nepochopitelné nejen pro potenciální uživatele, ale dokonce pro autory samotné. Bylo rozhodnuto, že se přejde k nové koncepci, kterou budou charakterizovat dva znaky: modularita a metabáze. Stalo se tak jednak proto, že se přes přílišnou složitost konceptu rodin veličin nechtělo úplně znemožnit jeho případné vzkříšení, jednak proto, že nová koncepce umožňovala podstatně jednodušší rozvíjení systému různými směry, také dle potřeb konkrétního uživatele.

Na obr. 1.4 je vidět, jakým způsobem byla navržena modularita nově vznikajícího systému (lze si všimnout souladu s metodikou CRISP-DM). Modul je v tomto případě samostatná aplikace, která je součástí celého systému a může být spouštěna a autory upravována nezávisle na ostatních modulech. Je zřejmé, že oproti původní konstrukci aplikace 4ft-Miner musela být interpretace výsledků oddělena od zadávání úlohy. Toto oddělení se provedlo rozdělením procedury 4ft-Miner na dva moduly: 4ftTask a 4ftResult. Pro každou z implementovaných procedur bylo pak rozdělení provedeno stejným způsobem: modul xxTask sloužil pro zadávání úlohy, generování a verifikaci ve fázi analýzy dat, modul xxResult pak pro prohlížení výsledků ve fázi jejich interpretace.

Velmi důležitou změnou bylo také zavedení dříve zmíněné metabáze, která byla centrálním úložištěm uživatelem zadaných dat a zároveň sloužila pro komunikaci mezi všemi moduly systému (viz obr. 1.5). Metabáze byla s analyzovanými daty asociována pomocí nově vytvořeného modulu LM Admin a toto asociování se stalo prvním krokem každé analýzy dat. Pro



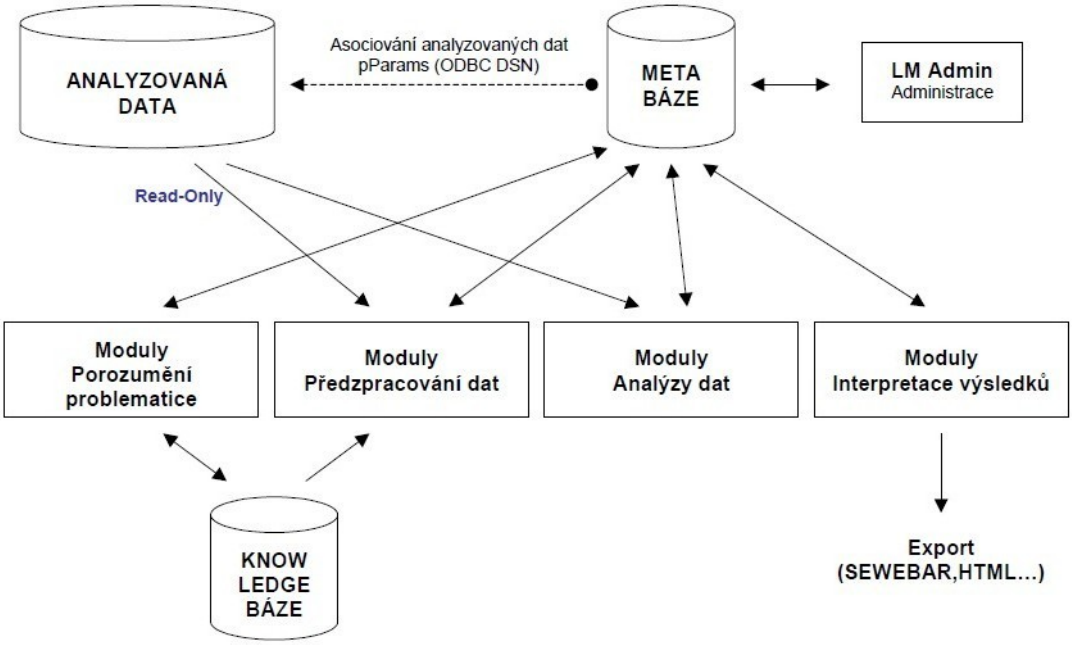
Obrázek 1.3: Objektový model kvantifikátorů z původní dokumentace 4ft-Mineru. Zdroj: [7]

metabázi byl zvolen databázový formát, což umožnilo používání metadat externími aplikacemi, ruční opravy, zálohování atd.

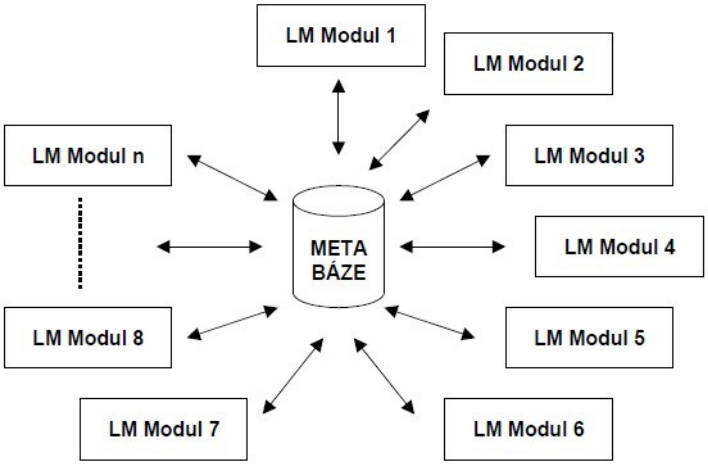
Novým prvkem oproti původní koncepci byla také knowledgebáze (aneb báze znalostí, později báze zkušeností – zvolený anglický název je analogický k metabázi). Knowledgebáze sloužila pro uchování obecně platných doménových znalostí (např. z oblasti lékařství), obsahovala i návrhy způsobu předzpracování dat (délky intervalů, prahové hodnoty apod.). Z hlediska celkové koncepce byla knowledgebáze součást metabáze, byla však vyčleněna jako zvláštní entita, protože mohla být vícenásobně používána mnoha metabázemi.

Změna konceptu na modulární a zavedení metabáze se ukázaly být krokem správným směrem a tento koncept se stal základem pro další vývoj systému, nově pojmenovaného LISp-Miner. Systém byl v roce 2011 tvořen těmito moduly:

1. Moduly pro porozumění problematice:
   - LAM LAQ Manager
     Formulace lokálních analytických otázek.
   - LM Knowledge Base
     Definice metaatributů (vhodná kategorizace, prahové hodnoty).
2. Moduly pro seznámení s daty a jejich předzpracování:
   - LM DataSource
     Příprava dat pro další zpracování: slučování sloupců, kategorizace, počítání četností, frekvenční a kontingenční analýza atd.



Obrázek 1.4: Konceptuální schéma systému LISp-Miner souboru modulů. Zdroj: [7]



Obrázek 1.5: Metabáze jako centrální úložiště metadat. Zdroj: [7]

- LM TimeTransf
  Příprava časových řad.

3. Moduly analýzy dat:

   - xxTask
     Zadávání úlohy, generování, verifikace.
   - xxGen
     Dávkové generování úloh.
   - xxGridGen
     Distribuovaný výpočet na gridu.

4. Moduly interpretace výsledků:

   - xxResult
     Prohlížení výsledků.
   - AR2NL
     Převod asociačních pravidel do přirozeného jazyka.
   - LM SwbExporter, LM SwbImporter
     Export údajů z databáze do textových formátů a opačný proces.

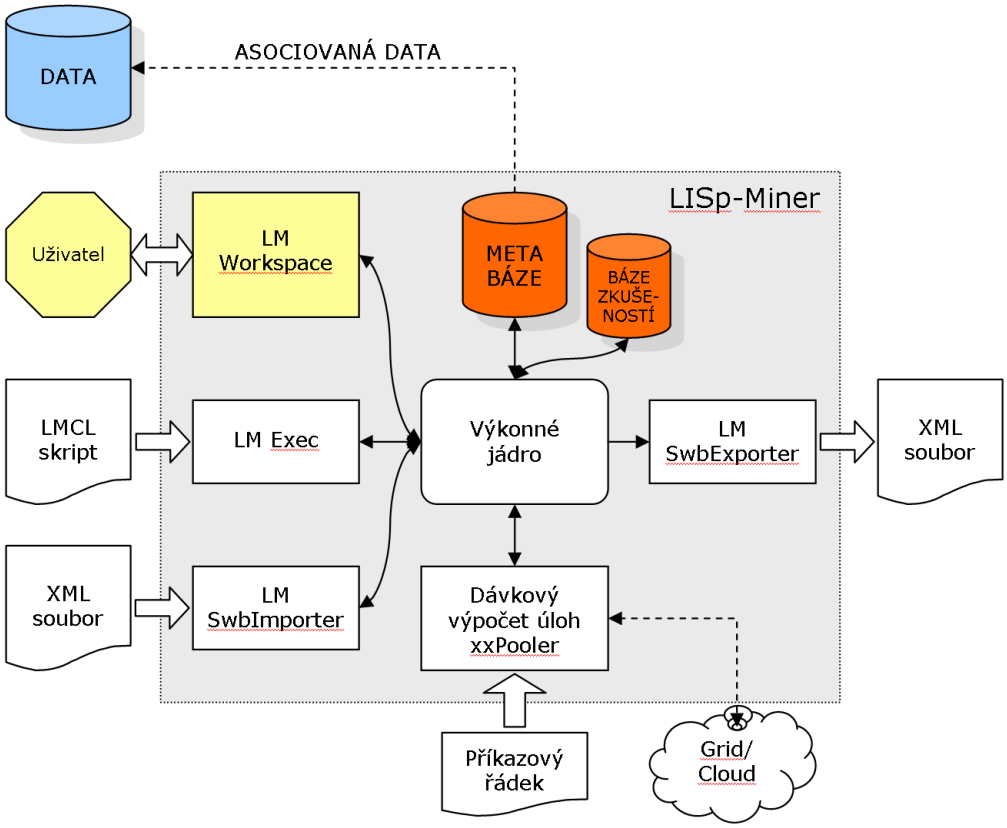
1.3.2 Současná podoba

Od přechodu na novou koncepci přibyla řada nových funkcí v existujících modulech, přibylo také několik nových procedur. Modul LM Admin zanikl a byl nahrazen modulem LM Workspace, do kterého byly dále integrovány moduly LM DataSource a především xxTask i xxResult, čímž přestaly být samostatnými aplikacemi. Na obr. 1.6 se nachází aktuální kontextový diagram systému.

Zde je potřeba ujasnit, že se pro přehlednost rozlišuje např. GUHA proceduru 4ft-Miner a modul 4ft-Miner. První je teoretický návrh procedury, druhá je její implementace v podobě modulu systému.

Kromě už popsaného modulu LM WorkSpace a výkonného jádra (jak je z obr. 1.6 zřejmé, společného pro celý systém) systém obsahuje další moduly, i nadále existující jako samostatné aplikace. Z důvodu odlišnosti jejich charakteru nebo určení k čistě samostatnému použití nejsou v kontextovém diagramu zobrazeny všechny. Seznam a stručný popis těchto modulů (převzatý z [7]) uvádím níže:

- moduly LM TaskPooler, LM ProcPooler a LM GridPooler
  Dávkový výpočet úloh na pozadí nebo na počítačovém gridu.
- ccGridGen
  Výpočet úloh na počítačovém gridu.



Obrázek 1.6: Aktuální kontextový diagram systému LISp-Miner. Zdroj: [7]

- LM Exec

  Spouštění skriptů v jazyce LMCL (Lisp Miner Control Language) – jazyce určeném pro skriptové ovládání LISp-Mineru (viz [7]).

- moduly LM SwbExporter a LM SwbImporter

  Výměna dat s jinými systémy.

- LM Reverse Miner

  Generování umělých dat.

- moduly KexTask a KexResult

  Procedura strojového učení Knowledge Explorer (KEx), zatím nebyly integrovány do modulu LM Workspace.

- LM LAQManager

  Podpora uživatele ve fázi porozumění doménové oblasti – pomáhá při formulaci lokálních analytických otázek; v současné době vývoj pozastaven.

- LM KnowledgeSource

  Podpora uživatele ve fázi předzpracování dat – umožňuje definovat metaatributy a k nim patřící informace o vhodné kategorizaci a prahových hodnotách; v současné době vývoj pozastaven.

1.3.3 Aplikace

Dosud nejdůležitější a nejrozsáhlejší využití našel systém LISp-Miner v oblasti medicíny, což se velmi blíží původní myšlence autorů metody GUHA. Jedním z výzkumných projektů, při kterých byl LISp-Miner použit, je český projekt STULONG (z ang. *LONGitudinal STUdy*) – longitudinální dvacetiletá studie rizikových faktorů aterosklerózy u mužů středního věku. Tato studie se věnovala hledání korelací mezi řadou parametrů popisujících samotnou nemoc a život nemocných za účelem nalezení případných způsobů prevence. Na souboru dat bylo definováno více než 60 atributů, které byly zpracovávány jednotlivými moduly systému LISp-Miner. Použity byly moduly 4ft-Miner, SD4ft-Miner, KL-Miner a KEx. Podrobnější popis průběhu výzkumu a jeho závěr najde čtenář v [14].

Dalším významným projektem z oblasti medicíny, ve kterém byl systém LISp-Miner použit pro výzkumné účely, je projekt SEWEBAR Tinnitus. Projekt SEWEBAR je jako takový zaměřen na prezentaci výsledků data miningu ve formě přijatelné pro nezkušené osoby (např. vlastníky dat, kteří nemají s data miningem žádný přímý styk), což se považuje za jeden z deseti základních problémů data miningu. Projekt SEWEBAR Tinnitus se zabývá hledáním závislostí, které by nastínily příčiny onemocnění tinnitem (šumem v uších). Projekt je v současné době ve fázi realizace. Krátký popis způsobu řešení, použitých dat a role systému LISp-Miner v celém projektu se nachází v [15].

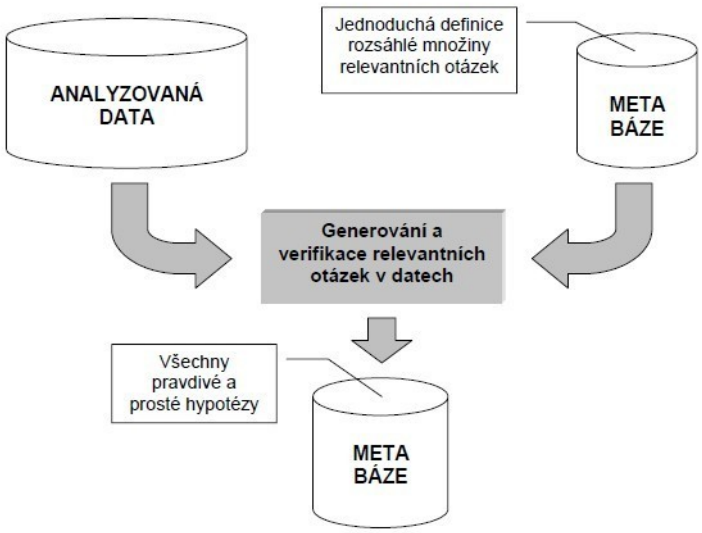
Předchůdce systému LISp-Miner, samostatný 4ft-Miner (viz 1.3.1, 2.1, byl také v minulosti použit v dobývání znalostí z dat týkajících se dopravních nehod ve Velké Británii. Tato aplikace (již zmíněná v úvodu) byla částí většího projektu a měla za účel nalezení asociačních pravidel s využitím kvantifikátoru AAD (viz tab. 2.1. V rámci zpracování byly hledány mj. konkrétní silnice, na kterých se určité typy nehod stávaly častěji. Tento výzkum je podrobněji popsán v [1]. Analogická úloha je zpracována i v rámci této diplomové práce (viz 5.1).

Kromě výše uvedených příkladů je LISp-Miner používán ve výuce předmětů týkajících se data miningu na několika vysokých školách, především na Vysoké škole ekonomické v Praze. Studenti, kteří během svého studia přišli s tímto systémem do styku, jej zkoušejí používat v různých případech a pro řešení různých problémů ve své kariérní dráze. Tyto pokusy a případné výsledky jsou bohužel neveřejné nebo se o nich ví jenom ze stručných zpráv od jejich autorů a nemohou být použity jako dobré příklady, přesto však naznačují, že LISp-Miner začíná být čím dál známější a díky svému modulárnímu charakteru může být použit v mnoha oblastech.

Kapitola 2

GUHA procedury aplikované v systému LISp-Miner

Jak už bylo řečeno v předchozí kapitole, systém LISp-Miner je založen na metodě GUHA a GUHA procedurách. LISp-Miner se vyznačuje modulární konstrukcí, přičemž většinou jeden modul odpovídá jedné GUHA proceduře. Implementováno bylo už mnoho modulů a program je stále vyvíjen. Tato kapitola se věnuje právě těmto modulům. Schéma GUHA procedury převedené do prostředí LISp-Mineru se nachází na obr. 2.1 (všimněme si analogie s obecným schématem GUHA procedury, které je umístěn v sekci 1.2 na obr. 1.2).



Obrázek 2.1: Konceptuální schéma GUHA procedury v prostředí LISp-Mineru. Zdroj: [7]

Každý modul pracuje se specifickým typem hypotézy. Každý takový typ se charakterizuje daným obecným tvarem a skládá se z cedentů nebo samotných atributů (někdy z nich obou). V další části kapitoly podrobněji popíšu tvar hypotéz hledaných jednotlivými procedurami a na příkladu uvedu vztahy, které daná hypotéza vyjadřuje (a které tedy lze zkoumat daným

modulem). Dále se každý modul vyznačuje používanými kvantifikátory. Tyto kvantifikátory se dělí na podtypy podle jedné či více vlastností. V závislosti na typu hypotézy se vztahují na souvislost mezi jejími jednotlivými součástmi nebo přímo na některou z těchto součástí. V další části kapitoly uvádím podrobnější výčet kvantifikátorů využívaných v jednotlivých procedurách včetně jejich popisu.

Kapitola se podrobně věnuje pouze třem základním modulům systému LISp-Miner, kterými 4ft-Miner, CF-Miner a KL-Miner. Tyto moduly jsou zároveň nejpoužívanější a postačí pro hledání typických závislostí, které lze v datech nalézt. Pro úplnost však uvádím v poslední sekci seznam všech ostatních současně implementovaných a funkčních modulů. V případě zájmu jejich podrobnější popis najde čtenář v [7].

2.1 4ft-Miner

4ft-Miner je nejstarší součást systému LISp-Miner a lze ho považovat za „původní“ implementaci metody GUHA v rámci tohoto systému. Tato procedura hledá závislosti v podobě 4ft-asociačních pravidel, která jsou ve tvaru

$$\text{antecedent} \approx \text{sukcedent},$$

nebo, pokud se jedná o podmíněná 4ft-asociační pravidla, ve tvaru

$$\text{antecedent} \approx \text{sukcedent} \;/\; \text{podmínka},$$

přičemž všechny cedenty jsou odvozené booleovské atributy.

Je zřejmé, že podmínka může být prázdná (jedná se tehdy o nepodmíněné pravidlo). Prázdný může být i antecedent, sukcedent však musí být tvořen alespoň jedním literálem, což je softwarově vynuceno (nelze zadat délku menší než jedna). Znamená to, že systém je schopen nalézat v datech i jednoduché charakteristiky, například fakt, že kvalita výrobku nikdy neklesá pod 90 % (pravidlo platí při prázdné podmínce a prázdném antecedentu, protože jednoduše platí vždy). Uvedu zde dobrý příklad (pocházející z [7]), který ilustruje, jaká pravidla hledá 4ft-Miner. Může to být pravidlo

$$\text{District(Praha, Plzen)} \wedge \text{Age}\langle 20, 30) \Rightarrow_{0,71;30} \text{Quality(bad)},$$

kde

- antecedent (předpoklad) je splněn, pokud klient banky **žije v Praze NEBO v Plzni A ZÁROVEŇ** je **ve věku mezi dvaceti** (včetně) **a třiceti** lety,
- sukcedent (závěr) tvrdí, že **kvalita splácení úvěru bude špatná**,
- antecedent a sukcedent jsou ve vztahu daným kvantifikátorem *fundovaná implikace* (viz tab 2.1) s hodnotou míry zajímavosti (*confidence*) 71 % a minimálním počtem takových případů v datech (BASE) rovným třicet (pro popis těchto parametrů viz dále).

Tabulka 2.1: Seznam funkčních kvantifikátorů procedury 4ft-Miner. Zdroj: [7]

Název	Zkr.	Poznámka
Support	SUPP	$a/(a+b+c+d) \geq p$ – alespoň $100\% \cdot p$ záznamů splňuje jak antecedent (A), tak sukcedent (S)
Founded Implication	FUI	$a/(a+b) \geq p$ – alespoň $100\% \cdot p$ záznamů splňujících A splňuje i S
Lower Critical Implication	LCI	Binomický test, který na úrovni α zavrhuje nulovou hypotézu $P(S\|A) \leq p$ ve prospěch alternativní hypotézy $P(S\|A) > p$
Upper Critical Implication	UCI	Binomický test, který na úrovni α přijímá nulovou hypotézu $P(S\|A) \leq p$ v neprospěch alternativní hypotézy $P(S\|A) > p$
Above Average Dependence	AAD	Mezi záznamy splňujícími A je alespoň o $100\% \cdot p$ více záznamů zároveň splňujících S než záznamů splňujících S v celé datové matici, resp. podmnožině matice dané aktuální podmínkou
Below Average Dependence	BAD	Mezi záznamy splňujícími A je alespoň o $100\% \cdot p$ méně záznamů zároveň splňujících S než záznamů splňujících S v celé datové matici, resp. podmnožině matice dané aktuální podmínkou
Double Founded Implication	DFUI	$a/(a+b+c) \geq p$ – alespoň $100\% \cdot p$ záznamů splňujících A nebo S splňuje zároveň A i S
Double Lower Critical Implication	DLCI	Binomický test, který na úrovni α zavrhuje nulovou hypotézu $P(A \wedge S\|A \vee S) \leq p$ ve prospěch alternativní hypotézy $P(A \wedge S\|A \vee S) > p$
Double Upper Critical Implication	DUCI	Binomický test, který na úrovni α přijímá nulovou hypotézu $P(A \wedge S\|A \vee S) \leq p$ ve prospěch alternativní hypotézy $P(A \wedge S\|A \vee S) > p$
Founded Equivalence	FUE	$(a+d)/n \geq p$ – alespoň $100\% \cdot p$ záznamů má stejnou pravdivostní hodnotu z hlediska platnosti A a S
Lower Critical Equivalence	LCE	Binomický test, který na úrovni α zavrhuje nul. hypotézu $P(A$ a S mají stejnou pravdivostní hodnotu$) \leq p$ ve prospěch alternativní hypotézy $P(A$ a S mají stejnou pravdivostní hodnotu$) > p$
Upper Critical Equivalence	UCE	Binomický test, který na úrovni α přijímá nul. hypotézu $P(A$ a S mají stejnou pravdivostní hodnotu$) \leq p$ v neprospěch alternativní hypotézy $P(A$ a S mají stejnou pravdivostní hodnotu$) > p$
Simple Deviation	SID	$a \cdot d > e^{\sigma} \cdot b \cdot c$ – alespoň $100\% \cdot p$ záznamů má stejnou pravdivostní hodnotu z hlediska platnosti A a S
Fisher quantifier	FSH	Fisherův test zamítající na úrovni α nulovou hypotézu nezávislosti A a S ve prospěch alternativní hypotézy jejich pozitivní logaritmické závislosti
Chi-Square quantifier	CHI	χ^2 test zamítající na úrovni α nulovou hypotézu nezávislosti A a S ve prospěch alternativní hypotézy jejich pozitivní logaritmické závislosti

Jednoduše řečeno toto pravidlo říká, že pokud klient žije v Praze nebo Plzni a je mladý, tak s vysokou pravděpodobností kvalita splácení jeho úvěru bude špatná. Pravidlo bude procedurou nalezeno, pokud tato tvrzení budou v datech pravdivá alespoň v 71 % případů a pokud zároveň absolutní počet takových záznamů bude větší nebo roven 30.

Cílem procedury 4ft-Miner je nalézt všechna taková pravidla, která v datech platí (samozřejmě s ohledem na použitý kvantifikátor a další parametry).

Používané kvantifikátory

Kvantifikátory se v proceduře 4ft-Miner dělí na funkcionální a agregační. První skupina obsahuje kvantifikátory, jejichž kritérium je funkcí hodnot z čtyřpolní tabulky a které slouží k posouzení „zajímavosti“ hypotézy. Druhou skupinu tvoří kvantifikátory založené přímo na hodnotách z čtyřpolní tabulky, popř. na jejich součtech. Ty se vztahují přímo na výskyt daných případů v datech. V případě druhé skupiny je možno zadat prahovou hodnotu absolutní nebo relativní, a to ve vztahu ke všem záznamům v datech, k záznamům splňujícím momentálně vygenerovanou podmínku hypotézy, nebo k nejvyšší hodnotě četnosti z čtyřpolní tabulky.

V tab. 2.1 uvádím seznam funkčních kvantifikátorů možných k použití v modulu 4ft-Miner. Pokračováním tohoto seznamu je seznam agregačních kvantifikátorů (tab. 2.2).

Tabulka 2.2: Seznam agregačních kvantifikátorů procedury 4ft-Miner. Zdroj: [7]

Název	Zkr.	Poznámka
Base	BASE	$a \geq$ BASE – alespoň BASE záznamů musí splňovat zároveň A i S, aby závislost byla statisticky relevantní
Ceiling	CEIL	$a \leq$ CEIL – ne víc než CEIL záznamů smí splňovat zároveň A i S (aby nešlo o příliš zjevnou závislost)
a-frequency	a	četnost a ze čtyřpolní kontingenční tabulky
b-frequency	b	četnost b ze čtyřpolní kontingenční tabulky
c-frequency	c	četnost c ze čtyřpolní kontingenční tabulky
d-frequency	d	četnost d ze čtyřpolní kontingenční tabulky
r-frequency	r	četnost $r = a + b$ ze čtyřpolní kontingenční tabulky
s-frequency	s	četnost $s = c + d$ ze čtyřpolní kontingenční tabulky
k-frequency	k	četnost $k = a + c$ ze čtyřpolní kontingenční tabulky
l-frequency	l	četnost $l = b + d$ ze čtyřpolní kontingenční tabulky
Sum of values	SUM	Součet všech hodnot čtyřpolní kontingenční tabulky (hodnota n)
Min value	MIN	Minimální četnost v kontingenční tabulce
Max value	MAX	Maximální četnost v kontingenční tabulce

2.2 CF-Miner

Procedura CF-Miner hledá v datech jiný typ závislostí. Cílem tohoto modulu je najít určitým způsobem zajímavá rozdělení frekvencí na nějaké podmnožině dat, kterou definuje aktuálně vygenerovaná podmínka. CF-hypotézy mají tvar

$$\text{atribut} \ / \ \text{podmínka},$$

přičemž podmínka je odvozený booleovský atribut (a může být prázdná). Příklad CF-hypotézy opět převezmu z [7]. Mějme hypotézu

$$\text{Age}_{\text{VarRatio}_{0,9;30}} \ / \ \text{District(Prague)}.$$

Podle této hypotézy:

- pro klienty **žijící v Praze**
- bylo pro kategorie atributu **Age** (věk)
- zjištěno **nahromadění četností v jedné kategorii** (ve smyslu daného kvantifikátoru) dané koeficientem 0,9 a s počtem takových případů (BASE) vyšším nebo rovným 30.

Zjednodušeně řečeno bylo při zkoumání četností jednotlivých definovaných věkových kategorií (nebo jejich kombinací) zjištěno, že v případě lidí žijících v Praze je četnost jedné konkrétní kategorie (či jejich kombinace) pozoruhodná za předpokladů definovaných kvantifikátorem.

Je vidět, že CF-Miner nepracuje už s asociačními pravidly, nýbrž s definovanou množinou atributů a jedním cedentem (ve tvaru odpovídajícím tvaru cedentu v proceduře 4ft-Miner), který představuje podmínku. I nadále je však cílem najít všechny hypotézy, které mají daný tvar a zároveň platí v analyzovaných datech.

Používané kvantifikátory

Seznam kvantifikátorů používaných procedurou CF-Miner se nachází v tab. 2.3. Kromě kvantifikátorů zmíněných v této tabulce se dále používají některé kvantifikátory popsané v sekci věnující se proceduře 4ft-Miner (2.1).

2.3 KL-Miner

Procedura KL-Miner je určena k hledání zajímavých rozdělení frekvencí dvou atributů. Toto se provádí pomocí tabulky četností o rozměrech obecně $K \times L$, kde K a L jsou počty kategorií (či jejich kombinací) daných atributů. Zároveň hodnoty v tabulce musí splňovat danou podmínku. KL-hypotézy mají tvar

$$\text{atribut}_K \times \text{atribut}_L \ / \ \text{podmínka},$$

přičemž podmínka je opět odvozený booleovský atribut (a opět může být prázdná).

V [7] je uveden takovýto příklad KL-hypotézy:

$$\text{Salary} \times_{\text{Kendall}_{0,2}} \text{Amount} \ / \ \text{District(Prague)}.$$

Podle této hypotézy

- pro klienty **z Prahy**
- byla pro společný výskyt kategorií atributů **Salary** (výše platu) a **Amount** (výše úvěru)
- zjištěna **funkční závislost ve smyslu Kendallova kvantifikátoru** (viz níže; čím vyšší plat, tím vyšší úvěr) daná koeficientem 0,2.

Zjednodušeně řečeno bylo v datech zjištěno, že v případě klientů z Prahy existuje závislost mezi výší platu a výší úvěru, a to taková, že je vyjádřena Kendallovým kvantifikátorem (neboli velmi zjednodušeně přímá úměra) s daným koeficientem.

I v tomto případě se nejedná o asociační pravidla. Podobně jako u CF-Mineru pracuje KL-Miner s (tentokrát dvěma) množinami atributů a jedním cedentem odpovídajícím svým tvarem cedentu z procedury 4ft-Miner. Úkolem modulu je jako u ostatních procedur nalezení všech hypotéz zadaného tvaru, které budou v datech platné.

Tabulka 2.3: Seznam kvantifikátorů procedury CF-Miner. Zdroj: [7]

Název	Zkr.	Poznámka
Sum of values	SUM	Součet frekvencí v daném rozmezí kategorií
Min value	MIN	Minimální frekvence v daném rozmezí kategorií
Max value	MAX	Maximální frekvence v daném rozmezí kategorií
Average value	AVG	Průměrná frekvence v daném rozmezí kategorií
Any value	ANY	Každá frekvence v daném rozmezí kategorií
Variation ratio	V	Variační poměr $1 - f_{Mo}$, kde $f_{Mo} = \max_k f_k$
Nominal variation (norm)	NVN	Nominální variace (norm) $(\sum_k f_k(1 - f_k)) \cdot K/(K - 1)$
Discrete ordinary variation (norm)	DOVN	Diskrétní ordinální variace (norm) $(2 \sum_k F_k(1 - F_k)) \cdot 2/(K - 1)$
Arithmetic average	AVGA	Aritmetický průměr kardinálních hodnot (které reprezentují kategorie)
Geometric average	AVGG	Geometrický průměr kardinálních hodnot
Variance	VARI	Rozptyl kardinálních hodnot
Standard deviation	STDEV	Směrodatná odchylka kardinálních hodnot
Skewness	SKEW	Šikmost rozdělení kardinálních hodnot
Asymmetry	ASYM	Asymetrický koeficient rozdělení kardinálních hodnot
Variation from pattern	VAR	Celkový rozptyl od daného vzoru
Steps-up	S-UP	Počet „schodů nahoru“ v rámci histogramu
Steps-down	S-DN	Počet „schodů dolů“ v rámci histogramu

Používané kvantifikátory

Seznam kvantifikátorů používaných procedurou KL-Miner se nachází v tab. 2.4.

Tabulka 2.4: Seznam kvantifikátorů procedury KL-Miner. Zdroj: [7]

Název	Zkr.	Poznámka
Sum of values	SUM	Součet frekvencí v dané podmnožině $K \times L$ tabulky
Min value	MIN	Minimální frekvence v dané podmnožině $K \times L$ tabulky
Max value	MAX	Maximální frekvence v dané podmnožině $K \times L$ tabulky
Average value	AVG	Průměrná frekvence v dané podmnožině $K \times L$ tabulky
Any value	ANY	Každá frekvence v dané podmnožině $K \times L$ tabulky
Variation from pattern	VAR	Celkový rozptyl od daného vzoru
Chi-Square test	ChiSq	χ^2 test podobnosti K a L – čím vyšší hodnota, tím více jsou K a L závislé
Function – sum of rows	FncS	Kritérium pro součet maximálních hodnot v řádcích $K \times L$ tabulky (hodnota v procentech)
Function – each row	FncR	Kritérium pro každý řádek $K \times L$ tabulky (hodnota v procentech)
Conditional entropy H(C|R)	H(C|R)	Podmíněná entropie sloupců podle řádků – čím nižší hodnota, tím vyšší závislost $\langle 0, \log_2 L \rangle$
Mutual information MI(R,C) normalized	MI(R,C)	Vzájemná závislost mezi řádky a sloupci – čím vyšší hodnota, tím vyšší závislost $\langle 0, 1 \rangle$
Inf. dependence ID(R,C)	ID(R,C)	Informační závislost mezi řádky a sloupci – čím vyšší hodnota, tím vyšší závislost $\langle 0, 1 \rangle$
Asymetric information coefficient AIC(R,C)	AIC(R,C)	Koeficient asymetrické závislosti θ -– čím vyšší hodnota, tím vyšší závislost $\langle 0, 1 \rangle$
Kendall's coefficient	KEND	Hodnota Kendallova koeficientu τ_B – čím je hodnota dále od nuly, tím jsou sloupce a řádky více závislé $\langle -1, 1 \rangle$

2.4 Další procedury

V této sekci uvádím výčet a krátký popis dalších procedur (implementovaných opět do jednotlivých modulů), kterých lze pomocí systému LISp-Miner využít. Podrobnější popis najde čtenář v [7].

MCluster-Miner

MCluster-Miner spojuje principy shlukové analýzy s metodou GUHA. Účelem procedury je nalézt (jako vždy u GUHA procedur) všechny varianty shlukování pro zadanou úlohu. Pomocí podmínky tvořené standardním cedentem lze také omezovat soubor dat, na kterém je prováděno shlukování.

SetDifference (SD)

SetDifference je třída procedur, jejichž přínos spočívá v porovnávání dvou hypotéz stejného tvaru, z nichž se však každá zaměřuje na jinou podmnožinu dat. Cíl může být dvojí: buď hledání rozdílů mezi těmito podmnožinami, nebo hledání takových podmnožin, ve kterých rozdíly existují. Množiny se stanovují pomocí odvozeného booleovského atributu.

Zmíněný postup lze aplikovat na všechny tři podrobně popsané v této práci procedury a hypotézy, se kterými pracují. Vznikají tak SD4ft-Miner, SDCF-Miner a SDKL-Miner porovnávající příslušně 4ft-hypotézy, CF-hypotézy a KL-hypotézy. Proto se hovoří o třídě procedur SDxx.

Ac4ft-Miner

Ac4ft-Miner hledá zajímavé dvojice asociačních pravidel, které představují změnu nebo akci. Jsou to pravidla, které mají neměnnou společnou část a liší se pouze koeficienty některých literálů v proměnné části. To znamená, že se hledají takové hypotézy, pro které změna jednoho z parametrů znamená konkrétní změnu jiného parametru.

Etree-Miner

Etree-Miner hledá zajímavé stromy. Za tímto účelem se používají explorační stromy. Ty se liší od rozhodovacích stromů tím, že v místě větvení lze pracovat s více atributy, ne pouze s jedním. Výsledkem je vygenerovaný les stromů.

KEx (strojové učení)

Kex je procedura strojového učení založena na asociačních pravidlech. Obsahuje algoritmus tvořící rozhodovací pravidla z mnohorozměrných kategoriálních dat. Účelem je tvorba báze znalostí.

Kapitola 3

Předzpracování reálných dat

Dobývání znalostí z databází je metodou založenou především na praktické práci. Bez ní nemá pojednávání o DZD přímý přínos z hlediska inženýrské práce. Po položení teoretického základu v podobě předcházejících kapitol proto následuje vlastní zpracování dat s využitím dříve popsaných programů a procedur. Zpracování sleduje zmíněnou v první kapitole metodiku CRISP-DM a její jednotlivé fáze. Vzhledem k charakteru této práce se vyskytují menší odlišnosti (např. chybí fáze aplikace výsledků).

Reálná data se jen výjimečně podaří získat v podobě vhodné k přímému použití. Téměř vždy je potřeba provést řadu úkonů předcházejících samotné zpracování. Mezi ně patří např. odstraňování či doplňování neúplných údajů, konverze datových typů či import do vhodného prostředí. Bez toho by výsledky byly neúplné, zavádějící nebo by vůbec nebylo možné je získat. Často se stává, že proces předzpracování dat je složitější, pracnější a delší než samotná práce, mnohdy také vyžaduje větší úsilí kvůli své nezajímavosti a pocitu řešení nevýznamných drobností, který se dostaví během práce. Jinak tomu nebylo ani mém případě. Nelze však tuto etapu podcenit, pokud je účelem dosažení dobrých a spolehlivých efektů.

Tato kapitola se věnuje získání, přípravě a předzpracování dat, která byla podkladem pro tuto práci. Nejdříve jsou popsána samotná data: zdroj jejich získání, obsah, struktura a forma, v jaké byla uložena. Dále jsou popsány nástroje použité k jejich přípravě a zpracování včetně zdůvodnění jejich volby. Následuje popis procesu čištění dat a jejich doplnění o odvozené hodnoty i parametry. V konečné části jsou popsány definování atributů a kategorizace dat v systému LISp-Miner.

Vlastní práce probíhala ve dvou etapách. V první nebyla brána v potaz polohová data, v druhé pak ano. Dále také, v souladu s popisem metodiky CRISP-DM uvedeným v první kapitole (v sekci 1.1), jednotlivé fáze neprobíhaly vždy chronologicky tak, jak je v této práci uvedeno. Mnohdy byly například jednotlivé atributy doplňovány po zjištění nových informací, dodatečné parametry byly odvozovány z existujících sloupců podle aktuální potřeby apod. Je to zcela přirozená skutečnost v případě explorační analýzy dat. Pro větší přehlednost jsou jednotlivé popisy prezentovány zde v uspořádané posloupnosti.

3.1 Popis použitých dat

Tato sekce popisuje dva soubory dat, které jsem použil během práce. První soubor jsou data, která se týkají dopravních nehod v ČR. Druhý soubor jsou geografická data ze serveru OpenStreetMaps. Tato data měla doplňkový charakter a sloužila k odvození některých nových parametrů pro záznamy dopravních nehod v prvním souboru dat.

3.1.1 Policejní databáze dopravních nehod v ČR

Databáze nehod byla získána od Policejního prezidia Policie ČR (o tato data může požádat kdokoliv). Tato databáze obsahuje všechny nehody zaznamenané policií (PČR) v letech 2007-2013. Jeden záznam o nehodě tvoří více než 60 hodnot jednotlivých parametrů, ke kterým patří mj. geografická poloha, příčina nehody, přesný čas a datum, charakteristiky zúčastněných vozidel či stav řidiče. Úplný seznam údajů zde neuvádím, jelikož jeho funkci plní v tomto případě seznam atributů definovaných v LISp-Mineru (viz 3.5) na tomto souboru dat. Tento seznam se nachází v příloze C. Některé údaje jsou citlivé (např. rodná čísla řidičů nebo registrační značky vozidel) nebo by umožňovaly jednoduchou identifikaci zúčastněných osob (např. věk ve spojení s polohou), proto byly policií odstraněny před poskytnutím obsahu databáze. Ostatní údaje byly zahrnuty v rámci definovaných atributů.

Původní data jsou uložena v elektronické podobě ve formátu .csv nebo .xls. Hodnoty jednotlivých parametrů jsou uloženy separátně ve sloupcích, popř. jsou odděleny čárkami (v případě formátu .csv). Data z jednoho roku tvoří několik souborů. K dispozici jsem měl také tato data převedená do formátu .txt a konsolidovaná do jednoho, popř. dvou souborů v případě každého roku. Data jsou velmi objemná, proto nejsou součástí příloh k této práci.

Pro práci jsem použil jenom data týkající se nehod na území Středočeského kraje. Tato data tvoří přibližně 100 000 řádků. Omezení jsem provedl především z důvodu časové a paměťové náročnosti zpracování celého souboru dat, který byl tvořen přibližně 800 000 řádky.

3.1.2 Geografická data ze serveru OpenStreetMaps

Data OpenStreetMaps (OSM) jsou volně přístupná na internetu mj. v podobě mapy (viz [16]). Obsahují údaje o všech existujících geografických objektech: budovách, ulicích, silničních tazích, řekách atd. Každý záznam je popsán geografickou polohou a několika parametry (tagy), které vyjadřují charakteristiky daného objektu. Veškerá data jsou ukládána na serveru i verifikována registrovanými uživateli a jsou volně dostupná.

Data přístupná přímo na serveru OSM nemohou být stahována ve velkých objemech. Pro tyto účely lze použít pravidelně aktualizovaný server Geofabrik ([17]), na kterém se nachází data agregovaná do celistvých souborů podle geografických celků. V případě ČR je to jeden soubor (dělení na kraje není dostupné). Data jsou dostupná v několika formátech. Vzhledem k přijaté metodě zpracování bylo nejlepší použít formát .shp (ESRI Shapefile).

3.2 Příprava softwarového prostředí

Pro zpracování dat jsem použil kombinaci tří programů. Jsou to PostgreSQL 9.3.4, Quantum GIS 2.6 Brighton a LISp-Miner 25.03.00. Dále jsem použil rozšíření PostGIS 2.1.

PostgreSQL je open source databázový server. Podporuje většinu datových typů standardu SQL:2008, je taky považován za nejrychlejší volně dostupný systém tohoto druhu. Je podporován unixovými operačními systémy a operačním systémem Windows. Pro práci s databází se používá jazyk SQL. Hlavními výhodami jsou volná dostupnost (open source) a možnost propojení s geografickými informačními systémy pomocí zásuvného modulu PostGIS. Oficiální webová stránka se nachází na adrese [18].

Quantum GIS (zkráceně QGIS) je open source geografický informační systém. Podobně jako PostgreSQL je podporován unixovými operačními systémy a operačním systémem Windows. Umožňuje tvorbu, vizualizaci, úpravy, analýzu a publikování geoprostorových informací. V rámci této práce byl použit za účelem získání informací o místě nehody ve vztahu k vybraným geografickým lokacím. Oficiální webová stránka se nachází na adrese [19].

LISp-Miner (zkráceně LM) je freeware určený pro dolování znalostí z databází. Jeho podrobný popis se nachází na začátku této práce v sekci 1.3. Oficiální stránka se nachází na adrese [13].

PostGIS je volně přístupný zásuvný modul do PostgreSQL. Tento modul podporuje geografické objekty a umožňuje doplňování tabulek o sloupce, ve kterých budou ve správném (čili srozumitelném pro geografické informační systémy, např. QGIS) formátu uložena geografická data, pomocí nových funkcí a klíčových slov. Stránka, ze které lze tento modul stáhnout, se nachází na adrese [20].

Při práci s databází nehod jsem měl k dispozici data popsaná v předchozí sekci (3.1). Prvním úkolem bylo převedení souborů .xls do formátu .csv a následně spojení těchto souborů do konsolidovaných výstupních .txt souborů. Obdržel jsem data už tímto způsobem upravená. Poté bylo potřeba provést načtení podle popisu uvedeného v přílohách v sekci A.1. Podrobný výpis použitých SQL dotazů se nachází v příloze D.

V další řadě bylo nutno propojit všechny zmíněné programy. Po propojení je centrálním bodem PostgreSQL, ke kterému jsou připojeny dva ostatní programy. Jak QGIS, tak LISp-Miner jsou napojeny na databázi, ze které čerpají informace potřebné pro jejich správné fungování. V případě QGISu jde také o doplňování do databáze nových dat.

Návody k přípravě prostředí se nachází v příloze A.

3.3 Čištění a úpravy dat

Jak bylo řečeno v úvodu k této kapitole, data jsou v okamžiku jejich získání či načtení do používaných nástrojů zřídkakdy ve stavu umožňujícím zpracování. Ani při tvorbě této práce tomu tak nebylo. Získaná data obsahovala mnoho nevhodně uložených informací, byla mnohdy uložena v nesprávném formátu a obsahovala chybné záznamy.

Pro větší přehlednost jsem rozdělil tuto sekci na podsekce odpovídající jednotlivým úpravám, jaké bylo nutno provést. Podobně jako v předchozí sekci jsou zde umístěny popisy provedených úkonů. Podrobný výpis SQL dotazů je opět umístěn v příloze D.

3.3.1 Sjednocení formátu identifikačního čísla

Identifikační číslo nehody by mělo být dvanáctimístné číslo uložené ve sloupci *p1*. V některých případech, ve kterých bylo číslo kraje (tedy první dvě číslice) menší než deset, bylo

identifikační číslo uloženo jako jedenáctimístné bez počáteční nuly. Bylo nutno doplnit toto číslo o počáteční nulu pro možnost dalšího jednoduchého zpracování.

Za tímto účelem jsem vytvořil sloupec *php1* typu *text*, který jsem v prvním kroce naplnil hodnotami ze sloupce *p1*. Následně ve všech případech, ve kterých byla délka řetězce znaků v tomto sloupci rovna 11, jsem na první místo záznamu připsal nulu.

3.3.2 Sjednocení formátu data konání nehody

Datum konání nehody bylo v databázi uvedeno pěti různými způsoby (ve sloupci *p2a* typu *text*). Vyskytující se formáty byly RRRR-MM-DD, DD.MM.RRRR, DD/MM/RRRR, MM/DD/RRRR a číslo odpovídající formátu, v jakém uchovává informaci o čase a datu Microsoft Excel. Bylo potřeba formát sjednotit a uložit do nového sloupce typu *date*.

Pro realizaci tohoto úkolu jsem nejdříve vytvořil sloupec *phdatum* typu *date*. Nejdříve jsem vybral data, ve kterých byl použit formát RRRR-MM-DD a uložil datum do nově vytvořeného sloupce jako datový typ *date*. Totéž jsem provedl pro formát DD.MM.RRRR. V případě formátů používajících znak „/“ bylo potřeba rozlišit, zda se jedná o formát DD/MM, či o formát MM/DD. Po rozpoznání jsem opět uložil takto zjištěné datum do sloupce *phdatum*. V poslední etapě bylo potřeba přepočítat excelovský formát dat na den, měsíc i rok a opět uložit do zmíněného sloupce.

3.3.3 Vyjmutí čísel kraje a okresu

Pro jednodušší zpracování jsem se rozhodl uložit čísla kraje a okresu do samostatných sloupců. Za tímto účelem jsem použil první dvě, resp. třetí a čtvrtou číslici identifikačního čísla nehody (tentokrát už z upraveného sloupce *php1*) a uložil je do samostatných sloupců *ku_kraj* a *ku_okres* typu *integer*.

3.3.4 Přidání sloupců s geometrickými daty

Polohové údaje byly v databázi uloženy ve dvou sloupcích typu *text*: x a y. Data nebyla ve vhodném datovém typu, někdy místo tečky byla uvedena čárka a navíc v některých případech byly hodnoty uloženy jako kladné místo jako záporné. Za účelem zpracování jsem vytvořil sloupce *phx* a *phy* typu *double precision*. První etapou předpřípravy bylo nalezení desetinných čárek, jejich nahrazení tečkami a uložení výsledků do nových sloupců. Následně bylo potřeba kladné hodnoty vynásobit číslem -1. Pro odstranění chybných záznamů jsem nastavil u záznamů, ve kterých hodnota ležela mimo přípustné intervaly, sloupce *phx* a *phy* jako NULL.

Další etapou bylo převedení vyčištěných polohových údajů do formátu kompatibilního s QGISem. K tomu jsem vytvořil pomocnou tabulku *spatial_ref_sys* a naplnil ji parametry odpovídajícími České republice. Poté jsem pomocí funkcí implementovaných v rámci PostGISu vytvořil sloupec *the_GeomKrovak* a naplnil ho příslušnými hodnotami vypočtenými na základě sloupců *phx* a *phy*.

PostGISovská funkce má jednu charakteristickou vlastnost: sloupec s geografickými údaji pojmenovává názvem v uvozovkách (*"the_GeomKrovak"*) a není možné to změnit v jejím rámci. Tahle skutečnost způsobuje potíž, protože PostgreSQL vždy vrací název tohoto

sloupce bez uvozovek, zatímco volat sloupec je potřeba vždy s uvozovkami. Z toho důvodu nastává při připojení LISp-Mineru k databázi chyba: LM získá z PostgreSQL název sloupce bez uvozovek, načež při pokusu o získání jeho hodnot vrátí PostgreSQL informaci, že sloupec (bez uvozovek) neexistuje. Pro správné fungování propojení LM-PostgreSQL je potřeba odstranit z názvu uvozovky. Příslušný SQL dotaz najde čtenář v příloze D.

3.3.5 Úprava času konání nehody

Čas konání nehody byl uveden ve formátu HHMM ve sloupci *p2b* typu *text*, přičemž hodnota „2560“ znamenala, že čas konání nehody nebyl zjištěn. Podobně jako v případě identifikačního čísla se v části případů nevyskytovaly nuly na začátku záznamu. Nejdříve jsem vytvořil sloupec *ku_time* typu *text* (uložení jako *integer* nemá reálný přínos), do kterého jsem zkopíroval všechny hodnoty z původního sloupce. Následně jsem přidal jednu (resp. dvě) nulu na začátek ve všech případech, ve kterých byly v záznamu tři (resp. dva) znaky. Dále jsem ještě uložil první dvě číslice z takto upraveného sloupce do nového sloupce *ku_hour* typu integer pro další zpracování v LISp-Mineru. Hodnotu „25“ jsem zahrnul do X-kategorie (viz 3.5.6).

3.3.6 Úprava roku výroby vozidla a sjednocení jeho formátu

Ve sloupci *p47* byl uložen rok výroby vozidla (datový typ *text*). Tento rok bylo potřeba převést na datový typ *integer* pro další zapracování a odstranit nevyhovující záznamy – v některých případech namísto chybějícího záznamu v případě nezjištění roku výroby byla ve sloupci uvedena hodnota „XX“ nebo dvě mezery. Dále byl v některých případech rok uveden pomocí posledních dvou číslic (např. „96“ pro rok 1996) nebo dokonce jedné číslice (např. „7“ pro rok 2007), což bylo nutno sjednotit.

Za tímto účelem jsem vytvořil nový sloupec *p47int* typu *integer*, do kterého jsem neuložil žádnou hodnotu, pokud ani původní sloupec *p47* neobsahoval žádnou hodnotu, nebo pokud touto hodnotou bylo „XX“ či dvě mezery. Pokud byla délka záznamu v původním sloupci rovna 1, přidal jsem před tuto hodnotu řetězec znaků „200“. Pokud byla délka rovna 2 a hodnota byla větší než 20, přidal jsem řetězec „19“. Pokud byla délka rovna 2 a hodnota byla menší než 20, přidal jsem řetězec „20“. Konverze do datového typu *integer* proběhla vždy až po přidání řetězce znaků před původní hodnotu.

3.3.7 Úprava dalších parametrů na datový typ *integer*

Několik parametrů uložených v databázi bylo potřeba převést na typ integer z důvodu jejich ryze číselného charakteru. K těmto parametrům patřily počet osob usmrcených, těžce raněných a lehce raněných (příslušně sloupce *p13a*, *p13b*, *p13c*), celková hmotná škoda (*p14*), počet zúčastněných vozidel (*p34*), číslo komunikace (*p37*; v tom případě ovšem není charakter ryze číselný) a škoda na vozidle (*p53*). Za tímto účelem jsem vytvořil sloupce *ku_p13a_int*, *ku_p13b_int*, *ku_p13c_int*, *ku_p14_int*, *p34int*, *p37int* a *p53int*, všechny typu *integer*, a naplnil je daty z příslušných původních sloupců. Kromě toho jsem ve sloupcích *ku_p14_int* a *p53int* vynásobil původní hodnoty číslem 100 (v původních datech byla uvedena částka ve stokorunách).

3.4 Doplnění odvozených sloupců

Některé údaje nebyly v databázi obsaženy, dají se však jednoduše odvodit z dat, která jsou v databázi přítomna. Část z nich je zajímavá a jejich zohlednění při generování hypotéz bylo lákavé. Z toho důvodu bylo potřeba tyto údaje do databáze doplnit. Významnou část této skupiny údajů tvoří geografické informace odvozené s využitím QGISu. Parametry odvozené od geografických dat představují hlavní přínos geografického informačního systému pro tuto práci. Jejich tvorba vyžaduje současného využití PostgreSQL i QGISu a znalosti v oblasti specifických PostGISovských funkcí.

V této sekci jsou opět popsány jenom úkony, které jsem provedl za účelem odvození nových sloupců. Podrobný výpis SQL dotazů se nachází v příloze D. Pokud se jedná o geografická data, v této sekci se nachází pouze výčet zvolených geografických údajů. Popis vlastního technického provedení se nachází v příloze B.

3.4.1 Suma počtů raněných a mrtvých

V databázi byly zvlášť uloženy počty lehce raněných, těžce raněných a mrtvých (*p13c*, *p13b*, *13a*). Pro lepší možnost posouzení celkových zdravotních následků nehody bylo vhodné jejich sečtení. Za tímto účelem jsem vytvořil nový sloupec *ku_p13sum* typu *integer* a vyplnil ho sumou hodnot ze sloupců *ku_p13a_int*, *ku_p13b_int* a *ku_p13c_int*. Kromě toho jsem vytvořil sloupec *ku_p13ab*, do kterého jsem uložil součet počtů mrtvých a těžce raněných.

3.4.2 Stáří vozidla

V databázi byly k dispozici data týkající se roku výroby vozidla a data, ke kterému k nehodě došlo. Z těchto dat je možné přibližně (nikoliv přesně) odvodit stáří vozidla v letech. Za tímto účelem jsem vytvořil nový sloupec *ku_vehicle_age* typu *integer* a naplnil ho věkem vozidla spočteným tak, že od roku, ve kterém k dané nehodě došlo (sloupec *phdatum*), byl odečten rok výroby vozidla (sloupec *p47int*) a takto získané číslo bylo zvětšeno o 1. V případě chybějícího záznamu o roku výroby nebyl věk vozidla spočten.

3.4.3 Vzdálenost od vybraných objektů typu *point*

Geografické údaje typu *point* posloužily k odvození vzdálenosti každé nehody od nejbližšího bodu daného typu. Přesný postup při odvození je popsán v příloze B. Ke zpracování jsem vybral tyto objekty:

- bary, hospody, restaurace apod.,
- čerpací stanice,
- nemocnice,
- policejní stanice,
- poštovní úřady,
- školy.

Ke každé nehodě byl přiřazen nejbližší objekt daného typu a jeho vzdálenost od místa nehody.

3.5 Definování atributů v LISp-Mineru

Dosavadní práce s databází v PostgreSQL spočívala v přímém upravování dat buď pomocí transformace stávajících dat, nebo tvorby zcela nového obsahu databáze. V případě QGISu se jednalo také o práci s daty pocházejícími přímo z databáze a odvození od nich příslušných dalších údajů. Explorační analýza v LISp-Mineru se v tomto ohledu značně liší od práce s předchozími dvěma programy. Veškeré úkony jsou prováděny s počty konkrétních hodnot (frekvencemi), tedy jakoby nad databází. K zásahu do jejího obsahu nijak během tohoto procesu nedochází. Je to spojeno s konceptem metabáze, podrobněji popsaným v sekci 1.3.1. Tento přístup je založen na definovaní atributů.

Atributy jsou uživatelem definované parametry pocházející či odvozené z dat uložených v databázi. Obvykle se definují na (či „nad“) jednom sloupci databáze, mohou však být definovány i na více sloupcích. V rámci atributu se definují kategorie, kterým odpovídají určité hodnoty příslušného sloupce. Těchto hodnot může být v rámci kategorie libovolně mnoho, mohou být také definovány pomocí intervalů (v případě numerických dat).

Zmíněné dříve frekvence se v LM počítají vždy jako počet dat (či jejich kombinací) patřících do dané kategorie. Jelikož je celá metoda GUHA založena na frekvencích (pro podrobnější popis viz 1.2.2), vhodné definování atributů a následná kategorizace jsou kritické pro dosažení kvalitních výsledků dolování z dat. Z toho důvodu je potřeba při vytváření atributů brát v potaz několik faktů. Základem práce jsou vždy data v databázi a datový typ sloupce, který chceme použít, nelze tak např. vytvořit intervalové kategorie na sloupci typu text (z toho důvodu jsem převáděl některé sloupce na typ *integer*, viz 3.3). Především však je při kategorizaci dat potřeba vždy provést rozvahu, jaká data máme k dispozici, jakým způsobem plánujeme daný atribut použít v definici zkoumaných závislosti a tedy do kolika i jakých kategorií je vhodné tato data rozdělit.

V této sekci popíšu vytváření atributů s důrazem na důležité aspekty, jaké je potřeba během ní zvážit. Na konci sekce stručně popíšu, jakým způsobem jsem atributy vytvořil během práce s daty o nehodách (s odkazy na dříve popsané aspekty). Podrobný seznam definovaných atributů se nachází v příloze C.

3.5.1 Počet atributů

Počet atributů se odvíjí od počtu sloupců v explorované databázi. Za předpokladu, že jsou všechna získaná data relevantní (tento předpoklad je ovšem málokdy splněn), by měl každému sloupci, někdy kromě primárního klíče, odpovídat alespoň jeden atribut. Neexistuje však žádné doporučení pro omezení počtu atributů, jaké definujeme, protože v rámci jedné dataminingové úlohy se vždy použijí pouze ty atributy, pomocí kterých zadáme cedenty. Někdy na jednom sloupci lze definovat více odlišně konstruovaných atributů (zejména s odlišným počtem kategorií). Každého atributu, jaký definujeme, můžeme využít k tomuto zadání, proto obecně platí, že čím více atributů nadefinujeme, tím více máme možností nalezení zajímavých závislostí.

3.5.2 Počet kategorií v rámci atributu

Většinou platí, že příliš mnoho kategorií velmi ztěžuje formulaci hypotéz kvůli přílišnému roztříštění dat. Následkem je snížení všech frekvencí pod úroveň umožňující hledání závislostí

založených na vysokém zastoupení daných hypotetických vztahů v databázi. Proto často chceme stanovit počet kategorií např. mezi 10 a 20.

Někdy však charakter dat dovoluje (či dokonce vyžaduje) definování velkého počtu kategorií pro smysluplné využití. Příkladem může být atribut *Road_ number*. V rámci tohoto atributu jsem definoval 1410 kategorií, a to tak, že každému číslu silnice vyskytujícímu se v datech odpovídá právě jedna kategorie. Jedním z důvodů takového rozdělení do kategorií je nemožnost smysluplné agregace čísel silnic do větších shluků. Lze poznamenat, že i pokud by to bylo možné, veškeré shluky by bylo možno nahradit atributem založeným na bázi parametrů, podle kterých bychom tyto shluky vytvářeli, čímž by veškeré shlukování postrádalo smysl. Kromě toho lze takto podrobné kategorizace s výhodou využít k hledání silnic, na nichž se specifické jevy vyskytují nadprůměrně často (jako např. v případě úlohy popsané v sekci 5.1). S rostoucí zkušeností si bude badatel situací, ve kterých lze upustit od dodržování menšího počtu kategorií, všímat častěji.

3.5.3 Spojování kategorií

Často se stává, že se v datech vyskytují odlišné hodnoty, které jsou však z hlediska daného atributu ekvivalentní. Často příčinou je nepřesné vyplnění hodnot daného sloupce, kvůli čemuž hodnoty pro člověka stejné jsou softwarem vnímány jako odlišné (typicky např. „0“ a chybně vyplněné „O“). V takovém případě je možno kategorie vytvářet ručně a zahrnovat do nich příslušné hodnoty, je to ovšem problematické, pokud je potřeba vytvořit jejich velký počet (např. 100). V takové situaci je lepší využít možnosti spojování kategorií.

LISp-Miner vždy nabízí automatizovanou kategorizaci dat v rámci atributu. V případě nominálních dat je to vždy přiřazení jedné hodnotě jedné kategorie. Po využití této možnosti lze zvolit kategorie (čili jednotlivé hodnoty), které považujeme za ekvivalentní, a spojit je pomocí funkce „Join“ do jedné kategorie. Ve veškerých dalších úlohách používajících daný atribut budou frekvence počítány ze všech hodnot, které byly do takto vytvořené kategorie zařazeny.

3.5.4 Několikanásobné definování atributů na jednom sloupci

Je potřeba si uvědomit, že jeden atribut je vhodné někdy rozdělit do kategorií různým způsobem pro použití v různých úlohách. Ne vždy je pro danou úlohu vyhovující stejně detailní rozdělení jako pro jinou, může proto být užitečné vytvoření několika atributů podle aktuálně potřebné podrobnosti. V závislosti na aktuálně zpracovávané úloze se pak vybere jeden z nich.

Dobrým příkladem je atribut *Accident_ cause*, který je rozdělen do 64 kategorií odpovídajících možným příčinám nehody stanoveným PČR. Toto rozdělení je velmi podrobné, vychází z vnitřních potřeb PČR a je koncipováno dvouúrovňově. První dělení je z hlediska široce pojaté příčiny nehody (rychlost jízdy, závada na vozidle atd.), druhé zpřesňuje tuto obecnou příčinu (např. nepřizpůsobení rychlosti technickému stavu vozovky či závada brzd). Lze tedy definovat atribut, který bude odpovídat podrobností druhému dělení (*Accident_ cause*), a atribut, který rozliší nehody pouze podle prvního dělení (*Accident_ cause_ less_ cat*). V závislosti na dané úloze se pak zvolí vhodná podrobnost dělení.

3.5.5 Nepravidelná délka intervalů

V případě tvorby intervalových kategorií pro číselná data LISp-Miner nabízí dvě automatizované metody: vytvoření intervalů stejné délky a intervalů o stejných frekvencích. Není vždy vhodné rozdělit atributy na ekvidistantní intervaly, proto lze také vytvořit intervaly podle představy uživatele. Před vlastním vytvářením je vždy dobré provést frekvenční analýzu daného sloupce a prozkoumat histogram hodnot. Vytvoření mnoha intervalů s mizivou či dokonce nulovou frekvencí zbytečně ztěžuje zpracování úloh, ve kterých je daný parametr použít. Vždy je záhodno zvážit možnost odstranění některých hodnot či využití principu, na kterém je postavena logaritmická škála.

V případě kategorií tvořených intervaly rovněž existuje možnost spojování několika intervalů do jedné kategorie. Pokud tyto intervaly se sebou sousedí, budou spojeny do jednoho intervalu. V opačném případě bude spojení fungovat tak, jak je to popsáno v sekci 3.5.3.

3.5.6 X-kategorie

Zajímavou funkcionalitou LISp-Mineru je možnost stanovení tzv. X-kategorie. Tato kategorie zahrnuje tzv. „outstanding values“, tedy hodnoty, které leží mimo oblast tvořenou skoro všemi ostatními hodnotami, nespolehlivě by ovlivňovaly výpočty a jsou s velkou pravděpodobností chybnými záznamy. Zařazením do X-kategorie jsou tyto záznamy vyřazeny z výpočtu při zpracování úloh, ve kterých se příslušný atribut vyskytuje, díky čemuž nejsou výsledné hypotézy ovlivněny nespolehlivými hodnotami.

Při rozhodnutí, zda a jak definovat X-kategorii, je potřeba vždy zvážit, jaká data je do ní vhodné zařadit a zda její zřízení neznehodnotí výsledky. Většinou platí, že pokud se ve sloupci vyskytují nevyplněné řádky (NULL), je vytvoření X-kategorie legitimní (takovéto případy LISp-Miner dokonce sám zařazuje do X-kategorie). Někdy se však může stát, že nevyplněná hodnota odpovídá ve skutečnosti jiné hodnotě, např. nule. Může se také stát, že hodnot spadajících do X-kategorie je většina (v této práci se tak stalo např. v případě parametru *Specific_ location*). V takovém případě některé nalezené hypotézy mohou být ve skutečnosti neplatné, protože platí v situaci vyřazení dat spadajících do X-kategorie, ale ve skutečnosti platí jenom ve velmi omezeném počtu případů. Úlohou badatele je vždy zkontrolovat, zda běžné postupy a zásady platí v daném případě, a případně provést příslušné úpravy.

3.5.7 Tvorba atributů během práce s databází nehod

V rámci předzpracování dat jsem v LISp-Mineru vytvořil 65 atributů. Tento počet je částečně odvozen od počtu parametrů stanovených v původní databázi PČR, je však také navýšen o několik atributů definovaných na tomtéž sloupci. Ve většině atributů jsem vytvořil kategorie přesně odpovídající původním hodnotám, v některých případech jsem však tyto kategorie upravil. V mnoha případech jsem byl také nucen spojovat kategorie z důvodů popsaných v 3.5.3.

Skupina atributů *Accident_ cause(...)* je vytvořena na základě úvahy analogické k úvaze zmíněné v 3.5.2 způsobem podrobněji popsaným v seznamu atributů (viz příloha C): kromě zmíněných dříve atributů jsem také definoval skupinu pěti atributů *Accident_ cause-Driving* až *Accident_ cause-Speed*. Tyto byly definovány za účelem hledání hypotéz týkajících se účasti

podrobných kategorií nehod v rámci jedné obecné kategorie. Pro žádné jiné účely nejsou tyto atributy vyhovující.

Při kategorizaci hodnot hmotné škody jsem vytvořil intervaly různé délky. V případě atributu *Financial_loss_at_vehicle* intervaly mají délku 1000 pro hodnoty menší než 10 000, délku 10 000 pro hodnoty menší než 100 000 a délku 100 000 pro větší hodnoty (pro hodnoty větší než 500 000 jsem vytvořil jednu společnou kategorii). V případě atributu „Financial_loss“ mají intervaly délku 10 000 pro hodnoty menší než 30 000, pro větší pak mají délku 100 000 (výjimkou je jedna kategorie pro hodnoty větší než 1 000 000). Toto vychází z úvahy popsané v 3.5.5.

K atributům *Solid_object_type* a *Specific_location* jsem definoval přidružené atributy *Solid_object_type_no_X* a *Specific_location_no_X*, ve kterých jsem nedefinoval X-kategorii. Bylo to způsobeno velkým počtem případů, ve kterých se žádná pevná překážka ani specifické místo nehody nevyskytovaly. U ostatních atributů jsem definoval X-kategorii podle hodnot vyskytujících se v příslušném sloupci. Toto jsem provedl na základě teoretické úvahy analogické k úvaze popsané v 3.5.6.

Kapitola 4

Praktické poznámky k práci se systémem LISp-Miner

Práce se softwarem určeným k dobývání znalostí z databází se liší od práce s většinou inženýrských programů. Jak bylo řečeno v první kapitole, data mining nespočívá v posuzování konkrétních hypotéz, nýbrž v hledání nových. Na první pohled se může zdát, že díky tomu neexistují žádná omezení či jasná pravidla, která je potřeba dodržovat během vlastního dolování. Pokud však máme zájem o dosažení kvalitních výsledků, je dobré se řídit několika obecnými zásadami.

Obsahem této kapitoly je popis těchto zásad a několik námětů k zamyšlení před první prací s LM. Tyto náměty nejsou sice kriticky důležité pro úspěšné dolování z dat, přesto však mohou výrazně zjednodušit a urychlit práci a proto je zařazuji před sekcemi věnujícími se samotným dataminingovým úlohám a jejich parametrům. Doporučuji čtenáři seznámit se s obsahem této kapitoly v navrženém pořadí, a to i v případě chybějících základních znalostí z oblasti fungování systému LISp-Miner a konstrukce dataminingových úloh.

4.1 Frekvenční analýza atributu

Přípravě každé úlohy by mělo předcházet provedení frekvenční analýzy atributů (pomocí nástroje o stejném názvu). Výstupem této analýzy je histogram všech kategorií daného atributu. Tento nástroj umožňuje uživateli rychlé seznámení s rozložením hodnot daného atributu a už tím ho často provokuje ke zkoumání jisté závislosti či naopak k upuštění od dosavadního záměru. Někdy dokonce i samotný histogram může poskytnout zajímavé poznatky ohledně zpracovávaného atributu. Díky frekvenční analýze si lze v datech všimnout pozoruhodných fenoménů. Histogram může mít zajímavý průběh (odpovídat tvarem Gaussově křivce, exponenciále apod.), vyjadřovat vztah mezi hodnotou atributu a frekvencí (zvláště u číselných atributů, např. přímá úměra), může taky upozorňovat na konkrétní hodnoty (píky, nulová frekvence). Všechny tyto skutečnosti je dobré znát před nastavováním parametrů dataminingové úlohy týkající se daného atributu.

Konkrétním příkladem vzatým ze zkušenosti s databází nehod je např. atribut *Accident_cause*, tedy příčina nehody. Ze skoro sta tisíc nehod zaznamenaných na území Středočeského kraje příčina téměř 20 % z nich byla stanovena jako nevěnování se jízdě řidičem.

Tato skutečnost je zajímavá sama o sobě, jelikož objektivně potvrzuje obecné mínění, že nejdůležitější je dávat pozor za volantem. Zdůvodňuje také zákaz používání mobilních telefonů za jízdy. Kromě toho však takto vysoká hodnota upozorňuje badatele, že aplikace některých kvantifikátorů nejspíše mnohdy bude zatížena nepřínosnými hodnotami. Lze očekávat, že pokud je frekvence pro hodnotu „nevěnoval se“ tak vysoká v celé množině dat, bude takto vysoká za většiny podmínek, proto hledání vysokých frekvencí v rámci tohoto atributu vždy vygeneruje hypotézu, ve které sukcedentem bude právě hodnota „nevěnoval se“. Všechny tyto hypotézy však budou nepřínosné, protože velké zastoupení hodnot nevyplyne z podmínky či předpokladu, nýbrž z toho, že takových nehod je obecně nejvíce. Naopak zajímavých výsledků by bylo možné dosáhnout při hledání situací, ve kterých je relativní počet takovýchto nehod mnohem nižší.

Rozšířenou verzí frekvenční analýzy je CF analýza, která se podobá proceduře CF-miner (viz 2.2). CF analýza nabízí kromě standardních funkcí frekvenční analýzy generování histogramu hodnot a jiných příbuzných grafů v závislosti na podmínce dané jiným atributem. Lze například pozorovat změnu histogramu hmotné škody v závislosti na značce automobilu („Je pravdou, že čím dražší auto, tím častější vyšší škoda, nebo naopak čím dražší auto, tím ohleduplněji jeho vlastník řídí a vysoká škoda proto není častá?“) či mnoho jiných vztahů. Tato funkce je užitečná také ve chvíli, když chce badatel ručně ověřit jednu konkrétní hypotézu namísto generování celé skupiny hypotéz týkajících se příslušných cedentů.

4.2 Čitelný popis úlohy

Základem efektivní práce je každou definovanou úlohu vždy čitelně popsat. Hypotézy jsou v LISp-Mineru spojeny s úlohou, které se týkají, proto srozumitelný popis umožňuje jednoduché nalezení dříve vygenerovaných výsledků. Při rozsáhlejších výzkumných projektech vhodné pojmenování umožní také rychlé ověření, zda daná úloha nebyla už dříve realizována (a pokud ano, tak s jakými parametry). Kolonka „Comment“ umožňuje uložení doplňujících údajů, které pak budou zobrazeny vedle názvu úlohy.

Kromě čitelného popisu je významnou součástí práce zařazování úloh do skupin. LISp-Miner zobrazuje seznam všech definovaných úloh, což v případě definování jejich velkého počtu brzy vede k nepřehlednosti a zpomaluje práci. V případě vytvoření skupin úloh lze zobrazit pouze úlohy patřící do dané skupiny, čímž se zobrazený seznam výrazně zkrátí.

V rámci své práce jsem použil tento vzor pojmenování úloh:

Atribut v antecedentu(délka dílčího cedentu)=>Atribut v sukcedentu(délka dílčího cedentu)/Atribut v podmínce(délka dílčího cedentu)

Pokud byla délka dílčího cedentu rovna 1, neuváděl jsem ji. Pokud byla stejná úloha definována víckrát a její jednotlivé instance se lišily pouze použitým kvantifikátorem či jeho parametrem, uvedl jsem tyto údaje v závorce na konci. Do komentáře k úloze jsem uvedl kvantifikátor a jeho parametr vždy. Skupiny atributů jsem vytvářel po vzoru názvu úlohy, např. „Cause=>Others“ byla skupina obsahující všechny úlohy, v kterých byla příčina nehody v antecedentu.

4.3 Klonování úloh

Velmi výhodnou metodou vytváření nových úloh je klonování. V případě vytváření zcela nové úlohy je potřeba vždy nastavit všechny cedenty a kvantifikátor (ten je vždy přednastaven v závislosti na typu úlohy). V případě využití možnosti klonování se přirozeně použijí všechna nastavení platná pro klonovanou úlohu. Díky tomu lze ušetřit čas, zvláště při definování sady podobných úloh. Je to také velmi dobrá funkce pro porovnávání výsledků úloh, které se lehce liší, např. pouze parametrem kvantifikátoru.

Ještě důležitějším důvodem pro použití klonování je skutečnost, že při jakékoliv úpravě parametrů úlohy smaže LISp-Miner všechny vygenerované hypotézy. Pokud je generování časově náročné (což se stává velmi často), může to znamenat ztrátu velkého množství času kvůli nutnosti opětovného generování předchozí úlohy, pokud se badatel rozhodne ještě jednou prozkoumat její výsledky. I v případě neúspěšných úloh není vhodné odstraňovat výsledné hypotézy, jelikož mohou se později ukázat vhodné nebo alespoň mohou sloužit jako známka, že daný pokus už byl proveden a nemá smysl se k němu vracet. Má to zvláště velký význam, pokud na projektu pracuje více lidí, viz následující sekce (4.4).

4.4 Zálohování a sdílení metabáze

Jak je popsáno v 1.3.1, kritickou částí LISp-Mineru je metabáze, ve které jsou uloženy informace o zpracovávaném projektu. Nachází se v ní data o definovaných atributech, definované úlohy, jejich výsledky atd. Jelikož je LISp-Miner stále vyvíjen, mohou ve výjimečných případech nastát situace, ve kterých může být obsah metabáze poškozen. Zvláště velké riziko je spojeno s úpravami „živé“ databáze mimo LISp-Miner, jako tomu bylo po celou dobu práce s databází nehod. Z těchto důvodů je velice vhodné pravidelně zálohovat metabázi projektu, která se většinou nachází v souboru .mdb ve složce „DBCon“ v adresáři LISp-Mineru.

Díky tomu, že je metabáze uložena v jednom souboru, může na jednom projektu pracovat více lidí, kteří budou tuto metabázi sdílet. Souběžná práce je problematická, ale postupné zpracování na více stanovištích je možné, pokud před zahájením práce bude lokální soubor metabáze nahrazen souborem získaným od jiné osoby. Tímto způsobem lze např. nahradit nepřítomného člena týmu, požádat někoho o kontrolu výsledků atd.

4.5 Volba GUHA procedury

Definování dataminingové úlohy je vždy potřeba začít volbou GUHA procedury, kterou budeme na data aplikovat. Každá procedura hledá v datech jiný typ závislostí, proto je potřeba se vždy předem rozhodnout, která z nich bude nejvhodnější pro daný účel. Teoretický popis vztahů hledaných jednotlivými procedurami se nachází v kapitole 2.

LISp-Miner nabízí možnost použití několika procedur. Nejčastěji používanou procedurou je 4ft-Miner, který hledá v datech asociační pravidla (viz 2.1). CF-Miner a KL-Miner jsou v jistém smyslu obecnější a pokročilejší verze 4ft-Mineru. Ostatní procedury mají poněkud užší oblast aplikace. Během zpracování údajů o nehodovosti jsem aplikoval především proceduru 4ft-Miner a několikrát i proceduru CF-Miner. Právě tyto dvě procedury doporučuji použít,

pokud se čtenář chystá pracovat s LISp-Minerem poprvé, protože vztahy, které hledají v datech, jsou nejjednodušší k pochopení. V další řadě je možno použít proceduru KL-Miner. Ostatní procedury jsou pokročilejší, specializované nebo mají zcela jinou oblast aplikace.

Vodítkem může být skutečnost, že procedura 4ft-Miner odpovídá na otázky, jejichž výsledkem jsou konkrétní hodnoty všech atributů zohledněných při konstrukci úlohy. Jinými slovy se tato procedura hodí k hledání odpovědí na otázku *Bude hodnota X ovlivněna/konkrétní, přijme-li Y nějakou konkrétní hodnotu?*, popř. otázku *Bude hodnota X ovlivněna/konkrétní, přijme-li Y nějakou konkrétní hodnotu, přičemž v potaz vezmeme jenom data splňující podmínku, podle které bude mít Z nějakou konkrétní hodnotu?* Kromě atributů mohou X, Y a Z reprezentovat také skupiny atributů. V oblasti dopravy příklady mohou být otázky *Bude hodnota denní intenzity provozu nadprůměrná v některém konkrétním dnu týdne?*, *Existuje nějaké konkrétní město, ve kterém alespoň polovina dopravních nehod se stává s účastí automobilů registrovaných v cizině?* či *Nachází se někde silnice, na které se v rámci jednoho konkrétního měsíce přes 50 % smrtelných nehod s účastí chodců, stává v noci?*

Procedura CF-Miner se oproti tomu zabývá celými histogramy daného atributu. Odpovídá na otázku *Bude histogram atributu X zajímavý, pokud vezmeme v potaz data, ve kterých Y přijme konkrétní hodnotu?* Příklady mohou být otázky *Bude histogram nehod v jednotlivých letech reprezentovat klesající (rostoucí) funkci, pokud vezmeme v potaz pouze nehody, které se konaly na dálnicích a ve kterých důvodem byl alkohol?* či *Existuje nějaká značka automobilu, pro kterou je rozložení nehod do jednotlivých dnů týdne přibližně rovnoměrné?*

4.6 Nastavení parametrů úlohy

Po volbě procedury je potřeba vhodným způsobem nastavit její parametry. Základním parametrem je použitý kvantifikátor. Pro každou proceduru existuje řada kvantifikátorů (viz tab. 2.1, 2.2, 2.3 a 2.4), které zásadním způsobem ovlivňují způsob posuzování vygenerovaných hypotéz, proto při jeho volbě je potřeba vždy dobře zvážit, jakých výsledků chceme dosáhnout. Pro první zkušenosti jsou vhodné zejména tři základní funkční kvantifikátory procedury 4ft-Miner, dále pak několik kvantifikátorů procedury CF-Miner. Druhou etapou je volba cedentů, do kterých je potřeba vhodně zvolit atributy, rozlišovat antecedent a podmínku (pro 4ft-Miner) a vhodně volit délku i způsob generování parciálních cedentů a celého cedentu.

4.6.1 Volba a nastavení kvantifikátoru

Nejzákladnějším kvantifikátorem procedury 4ft-Miner je kvantifikátor FUI. Tento kvantifikátor hledá v datech vztahy, ve kterých při splnění dané podmínky za předpokladu definovaného v antecedentu je závěr definovaný v sukcedentu splněn alespoň v definovaném parametrem kvantifikátoru (např. 0,7, tedy 70 %) relativním počtu případů. Konkrétní příklad vztahu a jeho popis se nachází v sekci 2.1.

Dalšími základními kvantifikátory jsou kvantifikátory AAD a jeho mutace BAD. Jejich cílem je hledání takových podmnožin dat (definovaných antecedentem a podmínkou), pro které platí, že nějaká kategorie definovaná sukcedentem má na nich větší (pro BAD menší)

relativní frekvenci, než má na množině všech dat. Tento kvantifikátor je velmi praktický v situacích, ve kterých nás zajímají jevy, které se nekonají často ve srovnání s ostatními, ale přesto mohou být ovlivněny jinými skutečnostmi.

V případě nehod je faktem, že nejčastější jsou nehody osobních automobilů. Tato skutečnost téměř znemožňuje použití základního kvantifikátoru FUI v případě umístění v sukcedentu atributu *Vehicle_type*, protože počet jiného typu nehod většinou nepřesáhne malé hodnoty, např. 20 %, což je třeba zohlednit při nastavování parametru kvantifikátoru. I v případě, že nastavíme nízkou hodnotu, bude potřeba odfiltrovat z výsledků hypotézy týkající se kategorie, která převažuje na celé množině dat. Další příčinou nevhodnosti kvantifikátoru FUI je velký počet kategorií mnoha atributů, v následku čehož jsou relativní frekvence v naprosté většině případů malé a nejsou potom nalezeny žádné hypotézy. Konečně nás v případě nehod primárně zajímají ne konkrétní převažující kategorie nehod, nýbrž spíše ovlivnění jejich standardního rozdělení – v rámci toho se přirozeně naleznou i situace, ve kterých bude nějaká kategorie převažovat (pokud nepřevažuje na celé množině dat, tak její relativní poměr bude mnohem větší a bude LISp-Minerem nalezen). Z těchto důvodů jsem kvantifikátor AAD (popř. BAD) používal v rámci svého zpracování databáze nehod nejčastěji.

V případě procedury CF-Miner je fungování kvantifikátorů trochu složitější vzhledem k tomu, že pracují s celým histogramem, nikoliv jenom s konkrétními hodnotami. Základní možností je hledání „schodů“, tedy monotónních posloupností, pomocí kvantifikátorů S-UP a S-DN (viz tab. 2.3). Tyto kvantifikátory lze použít např. pro hledání typů nehod, jejichž počet během několika let stoupal či naopak klesal. Je také možno hledat histogramy, ve kterých je rozptyl hodnot velký nebo naopak malý. Toto se realizuje pomocí kvantifikátoru MIN, který hledá hypotézy, ve kterých má minimální frekvence nějakou hodnotu, např. definovanou relativně vůči maximální frekvenci (MIN = 90%· MAX).

4.6.2 Nastavení cedentů

Při nastavování cedentů v proceduře 4ft-Miner je potřeba především vhodně zvolit atributy, které umístíme do antecedentu a podmínky. Na první pohled se může zdát, že alespoň při některých kvantifikátorech lze tyto cedenty mezi sebou libovolně vyměňovat. Není to však pravda – podmínka ovlivňuje také to, co se při výpočtu považuje za celou množinu dat. Zjednodušeně řečeno podmínka definuje data, se kterými se následně pracuje při generování hypotéz. Je to zvláště dobře vidět při použití kvantifikátoru AAD (popř. BAD), který, jak je napsáno výše, hledá podmnožiny, ve kterých se daná kombinace kategorií vyskytuje častěji než na celé množině dat. Pokud zadáme v podmínce nějaký atribut, jako celá množina bude v tomto okamžiku chápána množina dat splňujících aktuálně vygenerovanou podmínku, nikoliv všechna data, jaká jsou k dispozici.

Další otázkou je nastavení délky cedentů a parciálních cedentů. Cedenty jsou tvořeny jedním nebo několika parciálními cedenty, parciální cedenty jsou pak tvořeny literály (zvolenými atributy). Při generování hypotézy bude z každého atributu vybrán daný počet kategorií, které dohromady vytvoří literál. Počet literálů generovaných v rámci jednoho parciálního cedentu roste během zpracovávání úlohy od minimální povolené délky (obvykle 0) do maximální povolené délky (popř. maximální možné, např. pokud je počet zadaných atributů větší než maximální povolená délka cedentu), přičemž délka roste vždy až po vygenerování všech možných hypotéz při dodržení aktuální délky (analogické pravidlo platí pro počet kategorií

v literálu). Kromě nastavení délky jednotlivých parciálních cedentů lze také globálně nastavit délku celého cedentu, čímž se zajistí, že počet literálů v jeho rámci bude vždy v povoleném rozmezí (pokud je maximální povolená délka celého cedentu menší než suma maximálních povolených délek parciálních cedentů, zahrnuty budou všechny možnosti vygenerování tohoto počtu literálů, např. dva literály z prvního a tři z druhého parciálního cedentu, potom pak tři literály z prvního a dva z druhého).

Kromě nastavení délky lze také zvolit, zda budou jednotlivé literály v rámci parciálního cedentu v relaci konjunkce, nebo disjunkce. Jednotlivé parciální cedenty jsou vždy v relaci konjunkce. Dále lze také nastavit způsob volby kategorií do literálu. Nejčastěji používaným způsobem je „Subset“, což znamená vybírání libovolné kombinace z dané množiny kategorií. Jinými velmi užitečnými způsoby jsou „Sequence“ (sekvence několika sousedních kategorií) a „Cyclical sequence“ (totéž s tím, že se poslední kategorie bere jako sousedící s první). Poslední způsob má smysl zvláště při generování literálů tvořených několika dny následujícími po sobě v týdnu, měsíci či hodinami během dne.

4.6.3 Další parametry

Kromě zmíněných základních parametrů LISp-Mineru je možno nastavit i další parametry úlohy. Patří k nim mj. způsob práce s chybějícími hodnotami či maximální povolený počet hypotéz (omezení pro případy nevhodného nastavení parametrů úlohy, které by vedlo k nalezení příliš velkého počtu irelevantních hypotéz). Více informací o těchto parametrech najde čtenář v [7] a [8].

4.7 Filtrování a interpretace výsledků

Správné nastavení úlohy a vygenerování hypotéz je pouze polovina úspěchu. Vygenerované hypotézy je potřeba vždy posoudit a zamyslet se nad jejich významem. V případě mnoha z nich se může jednat o redundantní hypotézy, o hypotézy zcela nesmyslné či naopak o hypotézy zcela zřejmé a tím nezajímavé. Je potřeba brát v potaz zvláště hypotézy tvořící dohromady větší celek. Např. trojice hypotéz „Na silnici II/303 je víc nehod v pátek“, „Na silnici II/303 je víc nehod v sobotu“, a „Na silnici II/303 je víc nehod v neděli“ dohromady tvoří hypotézu „Na silnici II/303 je víc nehod o víkendu“, která by nejspíše byla nalezena, pokud by byla maximální délka příslušného literálu rovna 3, nikoliv 1.

Výsledky úlohy jsou prezentovány jako seznam hypotéz (jako např. na obr. 5.1). Každou hypotézu lze zobrazit v několika podobách, mj. jako čtyřpolní tabulku (pro proceduru 4ft-Miner) či textový soubor, dále lze jejich seznam vyexportovat např. do formátu HTML. V rámci seznamu je možné seřazovat výsledky podle hodnoty parametru kvantifikátoru, literálů v antecedentu, sukcedentu či podmínce. LISp-Miner nabízí také pokročilé nástroje pro filtraci, založené např. na parametrech hypotéz či syntaxi, o těchto nástrojích však v této práci nepojednávám. Zájemci mohou nalézt dodatečné informace v [7] a [8].

Při interpretaci hypotézy, o které si myslíme, že je blízká pravdě, je potřeba vždy vzít v potaz všechny detaily a vyvarovat se rychlým, jednoduchým závěrům založeným pouze na dílčí skutečnosti. Velká relativní frekvence nemusí nic znamenat, pokud je absolutní frekvence velmi nízká. Tvrzení, že je daný úsek nebezpečný pro autobusy, pokud bylo těchto nehod

relativně dvakrát víc než průměrně, avšak se tyto nehody staly jenom třikrát za sedm let, těžko může být považováno za pravdivé.

Jak bylo řečeno dříve, ve své práci používám především kvantifikátory AAD a BAD, které hledají nadprůměrná či podprůměrná relativní zastoupení záznamů daného typu v souboru dat. Při interpretaci hypotéz nalezených pomocí těchto kvantifikátorů je vždy potřeba být opatrným v činění závěrů a uvědomovat si, co plyne z toho, že jsou dané atributy v daném vztahu. Například z nižšího relativního počtu daných nehod může plynout, že se jich koná za daných podmínek víc, ale zároveň může být pravdou jenom to, že se všech nebo většiny ostatních nehod koná méně. Úkolem badatele je posoudit, jak moc pravděpodobný je každý z těchto závěrů, např. pomocí doplňujících dataminingových úloh.

Některé výsledné hypotézy mohou splňovat veškeré podmínky dané úlohou (včetně minimálního počtu záznamů, kterých se týkají), přesto však nemusí přinášet žádné smysluplné poznatky. Zvláště u hypotéz tvořených mnoha literály, tedy týkajících se úzkých souborů dat, je potřeba se zamyslet, zda je daná hypotéza pouze náhodným fenoménem, který se v datech vyskytuje, nebo je vskutku projevem nějaké zákonitosti. Na tom, k jakému závěru badatel dospěje, závisí konečná podoba a tím i kvalita výsledků.

V případě použití delších literálů (např. sekvence třech hodin za sebou) je nutno zohlednit překrývající se hypotézy. Pokud se budou ve výsledcích vyskytovat např. hypotézy pro hodiny 9–12, 10–13, 11–14, je zřejmé, že „shluk“ bude obsahovat hodiny 9–14. Není však jasné, pro kterou hodinu platí hledaná vlastnost v největší míře. Z toho důvodu může být vhodné opakovaně provést úlohu, ve které se nastaveními omezí výsledky pouze na daný typ hypotéz, navíc s jednoduchými literály. Na základě toho lze potom jednodušeji posoudit míru splnění dané úlohy každou hodinou (nebo jakoukoliv jinou kategorií).

Nelze uvést jednoznačně správná doporučení ohledně interpretace výsledků, jelikož ty závisí na zkušenosti badatele. Jen těžko lze očekávat, že je možno podat přesný a vždy správný návod na interpretaci hypotéz. Přesto několik výše uvedených pravidel (založených na situacích, s jakými jsem se během své práce setkal) může být pro čtenáře přínosné. Tento výčet si neklade za cíl být úplným či vždy spolehlivým návodem. Je možné, že v některých případech bude dokonce kontraproduktivní, je proto potřeba k němu přistupovat s rozumem a vlastním uvažováním.

Kapitola 5

Dolování znalostí z databáze silničních nehod

Teoretický úvod do problematiky dolování znalostí z dat, pochopení funkce metody GUHA, struktury systému LISp-Miner, popisu jednotlivých procedur i vztahů, jaké hledají, příprava prostředí, předzpracování dat a seznámení se se základními radami a pravidly, jakými je dobré se řídit při práci, tvořily nezbytnou přípravu před touto kapitolou. Struktura práce odpovídá tomu, že samotné hledání závislostí v datech je vyvrcholením celého dlouhého procesu přípravy jak badatele, tak nástrojů, které mu budou nezbytně potřebné. Po ukončení všech těchto etap lze konečně přistoupit k hlavní práci a získávat první výsledky.

V této poslední kapitole se čtenář seznámí s průběhem a vybranými výsledky skutečného hledání závislostí v databázi silničních nehod s využitím metody GUHA a systému LISp-Miner. Databáze nehod je rozsáhlým zdrojem mnoha ještě neobjevených poznatků, je proto dobrým materiálem, na kterém lze ukázat široké spektrum dataminingových úloh. Popis jejich definování, průběhu a výsledků je jednak nejlepší metodou prezentace samotné badatelské činnosti a způsobů, jakými ji lze realizovat, jednak zdrojem nápadů potřebných ke stanovení nových úloh a tím inspirací pro budoucí vlastní práci čtenáře. Toto je také hlavní náplň této poslední – a i stěžejní – kapitoly této práce.

Zde je potřeba poznamenat, že cílem této kapitoly není prezentovat kompletní výsledky dosavadního výzkumu. Počet jednotlivých výsledků je opravdu velký, není proto možné zde prezentovat všechny. Není účelné ani úplné vyjmenování zkoumaných vztahů, jelikož prezentace zkoumaných problémů bez výsledků nemá žádný význam, prezentace suchých výsledků bez doplňujících je grafů a komentářů je taktéž nepřínosná. Také samotný provedený výzkum není zdaleka úplný a nepokrývá ani všechny definované nad databází atributy, navíc byl proveden pouze pro data ze Středočeského kraje – ostatní kraje teprve čekají na zpracování.

Každá sekce této kapitoly se věnuje jednomu typu úlohy. V rámci sekce je popsána motivace vedoucí k prozkoumání daného problému a na jejím základě je stanovena analytická otázka vedoucí ke konkrétnímu definování úlohy. V další řadě jsou podrobně popsána všechna relevantní nastavení úlohy, především použitá procedura, atributy a kvantifikátory. Následně jsou vyjmenovány všechny výsledky (nebo, pokud to jejich charakter neumožňuje, alespoň jejich hlavní typy). V koncové části sekce jsou podrobně okomentovány a interpretovány vybrané výsledné hypotézy, v některých případech jsou naopak krátce popsány všechny výsledky.

Předposlední dvě sekce se věnují dodatečnému cíli této práce, tj. využití geografických informačních systémů v dolování znalostí z databází. Jejich struktura je stejná jako v případě ostatních sekcí, jsou však zaměřeny především na prezentování přínosu, jaký plyne z aplikace GIS. Kromě toho jsou pak pro čtenáře další inspirací při vymýšlení nových otázek, na které lze v datech hledat odpovědi.

5.1 Asociační pravidla I – pravidla bez podmínky

Náplní této úlohy bylo hledání vztahu mezi číslem silnice a typem vozidla. Inspirací pro tuto úlohu byl výzkum popsaný v [1]. Analytická otázka v tom případě zní:

> *Existují silnice, na kterých se nehody vozidel daného typu stávají neobyčejně často?*

Předpokládaným výsledkem bylo nalezení několika silnic, které díky svému průběhu jsou lákavé pro motocyklisty, což způsobí nadprůměrné zastoupení motocyklových nehod.

Použitá nastavení

Úloha byla spuštěna s těmito parametry:

- procedura: 4ft-Miner,
- antecedent: délka 2, atributy *Road_number* a *Road_category*, oba s maximálním počtem kategorií rovným 1,
- sukcedent: délka 1, jeden atribut *Vehicle_type* s maximálním počtem kategorií rovným 1,
- podmínka: prázdná,
- kvantifikátory: BASE 20, AAD 0,5.

To znamená, že jsou v datech hledány takové hypotézy, podle kterých na dané silnici je poměr nehod daného typu vozidla vůči všem nehodám na této silnici alespoň o 50 % větší, než je poměr všech nehod tohoto typu vozidla ke všem nehodám vůbec. Hypotéza se pokládá za relevantní, pokud je počet nehod daného typu vozidel na dané silnici roven alespoň 20.

Atribut *Road_category* je použit ze dvou důvodů. Zaprvé slouží pro rozlišení dálnic, rychlostních silnic a silnic první třídy, které mají stejné číslo (např. I/11 a D11) – číslo „11“ a kategorie „dálnice“ je jiný antecedent než číslo „11“ a kategorie „silnice 1. třídy“, generované hypotézy jsou proto odlišné. Zadruhé zajišťuje zohlednění ve výpočtu záznamů, ve kterých není uvedeno číslo silnice – takových záznamů je v datech téměř 30 000 (30 %). Pokud by nebyl v antecedentu i parametr kategorie silnice (ta je oproti číslu uvedena u každého záznamu), v následku zavedení X-kategorie pro atribut *Road_number* by nebyly záznamy bez uvedeného čísla silnice vůbec zohledněny. Všechny tyto záznamy jsou ve výpočtu brány jako nesplňující daný antecedent, díky čemuž hypotézy odpovídají pesimistické variantě („ostatní nehody se staly jinde a působí tak proti hypotéze“) a jsou méně zatíženy chybou plynoucí z neúplnosti dat.

Výsledky úlohy

Výsledkem úlohy je 33 hypotéz. Přehled všech hypotéz se nachází na obr. 5.1. Hypotézy jsou uvedeny ve tvaru

$$AvgDf\ \text{Road\_number}(X)\ \&\ \text{Road\_category}(Y) >\div< \text{Vehicle\_type}(Z),$$

kde X je číslo silnice, Y je kategorie silnice, Z je typ vozidla a $AvgDf$ je číselný parametr, kterého hodnota je rovna rozdílu poměru počtu nehod splňujících antecedent a sukcedent k počtu všech nehod splňujících antecedent a poměru počtu všech nehod splňujících sukcedent k počtu všech nehod vůbec (viz tab. 2.1, kvantifikátor AAD). Hodnota 3 znamená tedy o 300 % větší poměrné zastoupení nehod (čili 4x větší, nikoliv 3x větší). Hypotézy jsou seřazeny podle kategorie atributu v sukcedentu, v druhé řadě pak podle hodnoty parametru *AvgDf*.

Nr.	Id	AvgDf	Hypothesis
1	33	14.217	**Road_number**(*1027*) & **Road_category**(*3_trida*) >÷< **Vehicle_type**(*Motocykl*)
2	20	3.006	**Road_number**(*102*) & **Road_category**(*2_trida*) >÷< **Vehicle_type**(*Motocykl*)
3	24	2.889	**Road_number**(*116*) & **Road_category**(*2_trida*) >÷< **Vehicle_type**(*Motocykl*)
4	21	2.007	**Road_number**(*105*) & **Road_category**(*2_trida*) >÷< **Vehicle_type**(*Motocykl*)
5	11	1.610	**Road_number**(*9*) & **Road_category**(*1_trida*) >÷< **Vehicle_type**(*Motocykl*)
6	30	1.029	**Road_number**(*605*) & **Road_category**(*2_trida*) >÷< **Vehicle_type**(*Motocykl*)
7	23	0.752	**Road_number**(*114*) & **Road_category**(*2_trida*) >÷< **Vehicle_type**(*Motocykl*)
8	22	0.730	**Road_number**(*112*) & **Road_category**(*2_trida*) >÷< **Vehicle_type**(*Motocykl*)
9	29	0.687	**Road_number**(*603*) & **Road_category**(*2_trida*) >÷< **Vehicle_type**(*Motocykl*)
10	31	0.573	**Road_number**(*610*) & **Road_category**(*2_trida*) >÷< **Vehicle_type**(*Motocykl*)
11	9	1.663	**Road_number**(*8*) & **Road_category**(*Dalnice*) >÷< **Vehicle_type**(*Osob_auto_prives*)
12	6	1.590	**Road_number**(*5*) & **Road_category**(*Dalnice*) >÷< **Vehicle_type**(*Osob_auto_prives*)
13	4	1.021	**Road_number**(*3*) & **Road_category**(*1_trida*) >÷< **Vehicle_type**(*Osob_auto_prives*)
14	1	0.941	**Road_number**(*1*) & **Road_category**(*Dalnice*) >÷< **Vehicle_type**(*Osob_auto_prives*)
15	28	0.749	**Road_number**(*508*) & **Road_category**(*2_trida*) >÷< **Vehicle_type**(*Nakl_auto*)
16	12	0.577	**Road_number**(*11*) & **Road_category**(*Dalnice*) >÷< **Vehicle_type**(*Nakl_auto*)
17	7	1.407	**Road_number**(*5*) & **Road_category**(*Dalnice*) >÷< **Vehicle_type**(*Nakl_auto_prives*)
18	18	1.195	**Road_number**(*38*) & **Road_category**(*1_trida*) >÷< **Vehicle_type**(*Nakl_auto_prives*)
19	15	1.078	**Road_number**(*16*) & **Road_category**(*1_trida*) >÷< **Vehicle_type**(*Nakl_auto_prives*)
20	2	0.729	**Road_number**(*1*) & **Road_category**(*Dalnice*) >÷< **Vehicle_type**(*Nakl_auto_prives*)
21	13	0.709	**Road_number**(*11*) & **Road_category**(*Dalnice*) >÷< **Vehicle_type**(*Nakl_auto_prives*)
22	5	0.679	**Road_number**(*3*) & **Road_category**(*1_trida*) >÷< **Vehicle_type**(*Nakl_auto_prives*)
23	10	2.027	**Road_number**(*8*) & **Road_category**(*Dalnice*) >÷< **Vehicle_type**(*Nakl_auto_naves*)
24	8	1.806	**Road_number**(*5*) & **Road_category**(*Dalnice*) >÷< **Vehicle_type**(*Nakl_auto_naves*)
25	17	1.675	**Road_number**(*19*) & **Road_category**(*1_trida*) >÷< **Vehicle_type**(*Nakl_auto_naves*)
26	3	1.250	**Road_number**(*1*) & **Road_category**(*Dalnice*) >÷< **Vehicle_type**(*Nakl_auto_naves*)
27	16	1.227	**Road_number**(*16*) & **Road_category**(*1_trida*) >÷< **Vehicle_type**(*Nakl_auto_naves*)
28	14	1.130	**Road_number**(*11*) & **Road_category**(*Dalnice*) >÷< **Vehicle_type**(*Nakl_auto_naves*)
29	19	0.719	**Road_number**(*38*) & **Road_category**(*1_trida*) >÷< **Vehicle_type**(*Nakl_auto_naves*)
30	25	0.598	**Road_number**(*118*) & **Road_category**(*2_trida*) >÷< **Vehicle_type**(*Autobus*)
31	26	2.409	**Road_number**(*330*) & **Road_category**(*2_trida*) >÷< **Vehicle_type**(*Jizdni_kolo*)
32	32	1.463	**Road_number**(*611*) & **Road_category**(*2_trida*) >÷< **Vehicle_type**(*Jizdni_kolo*)
33	27	1.219	**Road_number**(*331*) & **Road_category**(*2_trida*) >÷< **Vehicle_type**(*Jizdni_kolo*)

Obrázek 5.1: Výsledné hypotézy pro úlohu Číslo silnice ⇒ Typ vozidla

Interpretace výsledků

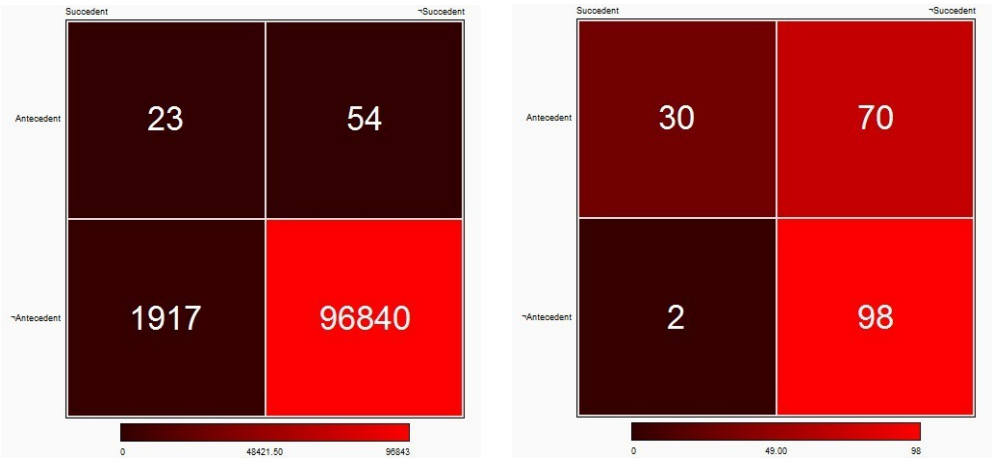
Vygenerovaných výsledků je velký počet a není zde prostor pro jejich úplnou interpretaci. Vybral jsem proto několik hypotéz a provedl jsem jejich podrobnější rozbor. Tento výběr má za účel znázornit možné typy výsledků a jejich interpretace, nikoliv pokrýt všechny vyskytující se typy a hypotézy.

Velice vysokou hodnotu *AvgDf* má hypotéza č. 1, podle které je poměrný počet motocyklových nehod na silnici III/1027 vyšší o 1 422 % (sic!) než průměr pro celý Středočeský kraj. Na obr. 5.2a jsou znázorněny absolutní počty záznamů, které splňují nebo nesplňují sukcedent či antecedent, na obr. 5.2b pak jejich relativní počty (pro každý řádek tabulky zvlášť). Je vidět, že až 30 % nehod jsou nehody motocyklů.

Pokud se podíváme do mapy (obr. 5.3), zjistíme, že silnice III/1027 vede podél vodní nádrže Slapy. Jedná se o malou silnici, na které je nejspíš provoz velmi klidný, proto je to zcela jistě zajímavá destinace pro motocyklisty. Nesmíme však činit příliš prudké závěry – za 7 let se těchto nehod konalo jenom 23, což znamená méně než jednu nehodu za čtvrt roku. Není to tedy silnice zvláště nebezpečná pro motocyklisty – bližší pravdě je spíše tvrzení, že se tam ostatních nehod koná velmi málo. Je z toho vidět, že ne vždy vysoký relativní počet daných nehod znamená nebezpečné místo, je proto potřeba být opatrným s vyvozováním důsledků.

Poněkud jinak se má situace v případě hypotézy č. 2, která se týká silnice II/102. Zde je relativní počet nehod několikrát nižší, avšak absolutní počet je dvakrát vyšší. Silnice II/102 vede z Prahy do Milevska, při pohledu do mapy není vidět nic zvláště lákavého pro motocyklisty. Přesto jsou tam však nehody motocyklistů relativně 4x četnější než je průměr (viz obr. 5.4). Může to znamenat nebezpečná místa, která by bylo záhodno upravit či zabezpečit. Stálo by za to zvážit, zda není potřeba tuto silnici podrobněji prozkoumat z provozně-infrastrukturního hlediska.

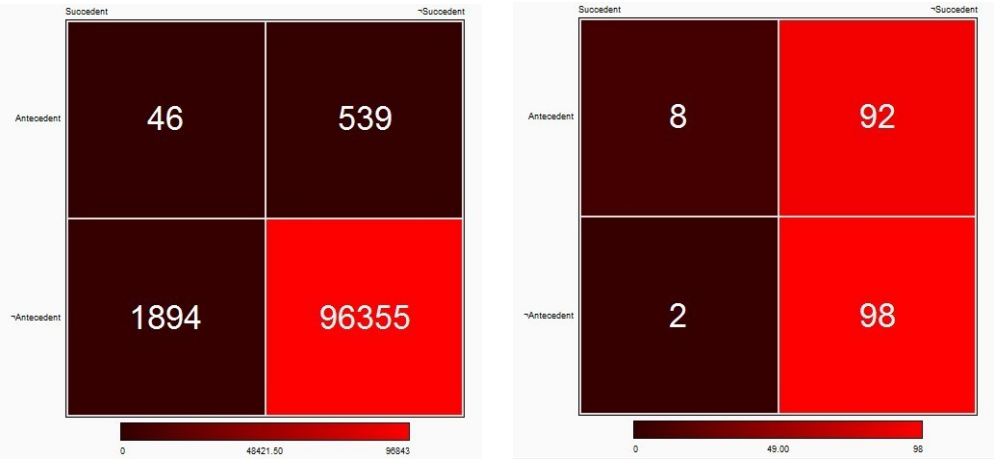
Další zajímavou hypotézou je hypotéza č. 31. Podle ní je relativní počet nehod jízdních



a) absolutní frekvence b) relativní frekvence

Obrázek 5.2: Čtyřpolní tabulky pro hypotézu č. 1 v úloze Číslo silnice ⇒ Typ vozidla.

Obrázek 5.3: Vyznačení silnice III/1027 (výsledku hypotézy č. 1) na mapě. Zdroj: [21]



a) absolutní frekvence b) relativní frekvence

Obrázek 5.4: Čtyřpolní tabulky pro hypotézu č. 2 v úloze Číslo silnice ⇒ Typ vozidla.

kol na silnici II/330 Český Brod – Činěves až o 241 % vyšší než je průměr. Absolutní počet nehod je roven 23, což je stejný počet jako v případě hypotézy č. 1, v případě cyklistů se však jedná o mnohem větší nebezpečí (lze očekávat, že cyklisté jsou mnohem zranitelnější kvůli odlišnému charakteru jejich účasti v silničním provozu).

Na první pohled to může znamenat pro cyklisty velmi nebezpečnou silnici, u které by stálo za to zvážit výstavbu cyklostezky. Nasvědčovat by tomu mohla také skutečnost, že výjezd z dálnice D11, který obsluhuje Český Brod a Sadskou, vede právě na tuto silnici. Pokud bychom se však podívali do mapy, zjistili bychom, že se valná většina těchto nehod konala ve městě, konkrétněji v Nymburce a Sadské. Důvodem je nejspíše fakt, že tato silnice tvoří v obou těchto městech jeden z hlavních tahů. Může to tedy znamenat, že u těchto silnic je vskutku dobré zvážit výstavbu cyklostezky, avšak především na území Nymburka a Sadské.

Poslední hypotézou, jakou zde popíšu, je hypotéza č. 30. Podle ní se na silnici II/118 konalo relativně o 60 % víc nehod autobusů než průměr. Absolutní počet nehod je roven 26, což v případě autobusů je velmi mnoho (autobusy jsou jedním z nejméně zastoupených typů vozidel v oblasti nehod). Silnice II/118 vede z Příbrami do Doksan, na první pohled není vidět žádný důvod, pro který by byl počet autobusových nehod vyšší. Zjištění přesné příčiny tohoto stavu je vzhledem k tomu, že se jedná o jedinou silnici ve Středočeském kraji s tak vysokým poměrným zastoupením tohoto typu nehod, velmi doporučené.

5.2 Asociační pravidla II – pravidla s podmínkou

Náplní této úlohy bylo hledání vztahu mezi charakteristikou vozidla a časem konání nehody (hodina, den týdne) za volitelné podmínky v podobě stavu řidiče (připomínám, že podmínka znamená v tom případě omezení celé množiny dat pouze na záznamy, které tuto podmínku splňují). Analytická otázka pro tuto úlohu zní:

> *Existuje nějaká charakteristika vozidla, pro kterou se nehody v daném čase nebo dni konají neobyčejně často, popř. neobyčejně vzácně? A totéž, pokud vezmeme v potaz pouze nehody, při kterých byl řidič v konkrétním stavu?*

Pro tuto otázku nebyly stanoveny žádné předem očekávané výsledky.

Použitá nastavení

Úloha byla spuštěna s těmito parametry:

- procedura: 4ft-Miner,
- antecedent: délka 1, atribut *Vehicle_ characteristics*,
- sukcedent: délka 1, dva atributy *Hour* a *Day_ of_ Week* s maximálním povoleným počtem kategorií rovným 1,
- podmínka: délka 0 až 1, atribut *Driver_ state* s maximálním povoleným počtem kategorií rovným 1,
- kvantifikátory: BASE 20, AAD 0,5 (popř. BAD 0,3).

To znamená, že jsou v datech hledány takové hypotézy, podle kterých na dané silnici je poměr nehod daného typu vozidla vůči všem nehodám na této silnici alespoň o 50 % větší, než je poměr všech nehod tohoto typu vozidla ke všem nehodám vůbec. Hypotéza se považuje za relevantní, pokud je počet nehod daného typu vozidel na dané silnici roven alespoň 20.

To znamená, že jsou v datech hledány takové hypotézy, podle kterých je poměr nehod vozidla dané charakteristiky stávajících se v daný čas vůči všem nehodám stávajícím se v tuto dobu o 50 % větší (popř. alespoň o 30 % menší), než je poměr všech nehod tohoto typu vozidel ke všem nehodám vůbec. Hypotéza se považuje za relevantní, pokud je počet nehod vozidla dané charakteristiky v daném časem či dni roven alespoň 20.

Výsledky úloh

Výsledky úloh je v případě úlohy s kvantifikátorem AAD 32 hypotéz, v případě kvantifikátoru BAD sedm hypotéz. Přehled všech hypotéz se nachází na obr. 5.5. Hypotézy jsou uvedeny ve tvaru

$$AvgDf \text{ Vehicle\_characteristics}(X) >\div< \text{Hour}(Y) \text{ (popř. Week}(Y)),$$

pokud se jedná o pravidlo bez podmínky, nebo

$$AvgDf \text{ Vehicle\_characteristics}(X) >\div< \text{Hour}(Y) \text{ (popř. Week}(Y)) \text{ / Driver\_state}(Z),$$

pokud se jedná o pravidlo s podmínkou. X je charakteristika vozidla, Y hodina (popř. den v týdnu) konání nehody, Z je stav řidiče a *AvgDf* je číselný parametr (pro popis viz 5.1). Hypotézy jsou seřazeny podle antecedentu, v další řadě pak podle sukcedentu a následně podle hodnoty *AvgDf*.

Interpretace výsledků

Tentokrát provedu interpretaci (nebo alespoň komentář) všech hypotéz, avšak méně podrobným způsobem oproti úloze popsané v předchozí sekci 5.1. Ttentokrát se nezaměřím na hodnotu parametru AvgDf, nýbrž jenom na skutečnost, že tento parametr překročil nastavenou mez. Zatímco metoda interpretace použitá v předchozí úloze měla za účel podrobné znázornění čtenáři smyslu jednotlivých procedur, v této úloze se jedná spíše o ukázku výsledků možných k dosažení pomocí dataminingového softwaru a o poukázání na některé zajímavé fenomény.

Nejdříve se budu věnovat interpretaci hypotéz vygenerovaných použitím kvantifikátoru AAD (obr. 5.5a). Hypotéza č. 1 říká, že pokud vezmeme v potaz pouze záznamy týkající se nehod, při kterých řidič byl unaven nebo usnul, tak soukromá vozidla nevyužívaná k výdělečné činnosti nejčastěji[1] havarují po 15. hodině. Můžeme z toho učinit závěr, po celém dni v práci (soukromá vozidla v tuhle dobu patří často právě lidem vracejícím se z práce) jsou lidé unaveni a mají nejvyšší tendenci způsobovat nehody. Rozšířený názor, že je dojíždění do práce autem rizikové, má tedy v sobě zrnko pravdy.

Podle hypotézy č. 2 mezi nehodami, při kterých byl řidič v dobrém stavu (neunaven apod.), skutečnost, že bylo soukromé vozidlo využíváno ve výdělečné činnosti, měla největší vliv na relativní počet nehod ve tři hodiny ráno. Nelze v žádném případě říci, že se nehody těchto vozidel stávají nejčastěji v tuto dobu – to je nesmysl. Zdůvodněním nalezení této hypotézy je spíše skutečnost, že ve 3 hodiny ráno jezdí primárně ti, kteří podnikají v oblasti dopravy. „Obyčejní“ občané v tu dobu spí, osoby zaměstnané ve firmách a používající jejich služební auta mají bezpečnostní předpisy, které musí dodržovat, ale např. živnostníci s jedním nákladním autem jezdí co nejvíce, aby co nejvíce vydělali. Není to však zdaleka spolehlivé zdůvodnění a tuto hypotézu nepovažuji za důvěryhodnou.

Hypotézy č. 3–4 říkají, že pokud vezmeme v potaz množinu nehod, při kterých byl řidič pod vlivem alkoholu, tak automobily patřící soukromým firmám a organizacím havarují nejčastěji po 7. a zvláště 8. hodině ranní. Interpretace je celkem jednoduchá: řidiči služebních aut často nemají na vybranou a pokud večer strávili s alkoholem, ráno stejně musí řídit (oproti uživatelům soukromých aut, kteří mohou zvolit veřejnou dopravu), což se projevuje zvýšeným počtem nehod.

[1] V tomto případě se ve skutečnosti jedná o pojem „relativně nejčastěji“, tzn. relativní počet nehod těchto vozidel stávajících se po 15. hodině se nejvíce liší od relativního počtu všech nehod stávajících se ve stejnou dobu. Musíme předpokládat existenci jistého rozdělení nehod do jednotlivých hodin – potom zjednodušeně řečeno v 15 hodin je vliv parametru „Charakteristika vozidla“ největší, tudíž tato vozidla „nejčastěji“ havarují právě v tuto dobu. Tento význam je spojen s použitím kvantifikátoru AAD, jehož funkci jsem podrobněji popsal v sekci 4.6.1. Tato poznámka se týká všech dalších interpretací hypotéz získaných pomocí kvantifikátoru AAD (popř. BAD).

Nr.	Id	AvgDf	Hypothesis
1	29	0.306	**Vehicle_characteristics**(*Soukr_nevydel*) >=< **Hour**(*15*) / **Driver_state**(*Unaven_usnul*)
2	15	0.502	**Vehicle_characteristics**(*Soukr_vydel*) >=< **Hour**(*3*) / **Driver_state**(*OK*)
3	30	0.833	**Vehicle_characteristics**(*Soukr_org_podnik*) >=< **Hour**(*7*) / **Driver_state**(*Vliv_alkoholu*)
4	31	1.440	**Vehicle_characteristics**(*Soukr_org_podnik*) >=< **Hour**(*8*) / **Driver_state**(*Vliv_alkoholu*)
5	32	0.328	**Vehicle_characteristics**(*Soukr_org_podnik*) >=< **Day_of_Week**(*Thu*) / **Driver_state**(*Vliv_alkoholu*)
6	1	0.405	**Vehicle_characteristics**(*VHD*) >=< **Day_of_Week**(*Mon*)
7	16	0.314	**Vehicle_characteristics**(*VHD*) >=< **Day_of_Week**(*Mon*) / **Driver_state**(*OK*)
8	2	0.843	**Vehicle_characteristics**(*Statni_pod_org*) >=< **Hour**(*6*)
9	3	0.400	**Vehicle_characteristics**(*Statni_pod_org*) >=< **Hour**(*7*)
10	4	0.514	**Vehicle_characteristics**(*Statni_pod_org*) >=< **Hour**(*8*)
11	17	0.402	**Vehicle_characteristics**(*Statni_pod_org*) >=< **Hour**(*8*) / **Driver_state**(*OK*)
12	5	0.866	**Vehicle_characteristics**(*Statni_pod_org*) >=< **Hour**(*9*)
13	18	0.771	**Vehicle_characteristics**(*Statni_pod_org*) >=< **Hour**(*9*) / **Driver_state**(*OK*)
14	6	0.417	**Vehicle_characteristics**(*Statni_pod_org*) >=< **Hour**(*10*)
15	19	0.382	**Vehicle_characteristics**(*Statni_pod_org*) >=< **Hour**(*10*) / **Driver_state**(*OK*)
16	20	1.398	**Vehicle_characteristics**(*Mimo_CR*) >=< **Hour**(*0*) / **Driver_state**(*OK*)
17	7	0.996	**Vehicle_characteristics**(*Mimo_CR*) >=< **Hour**(*0*)
18	21	2.076	**Vehicle_characteristics**(*Mimo_CR*) >=< **Hour**(*1*) / **Driver_state**(*OK*)
19	8	1.390	**Vehicle_characteristics**(*Mimo_CR*) >=< **Hour**(*1*)
20	22	1.893	**Vehicle_characteristics**(*Mimo_CR*) >=< **Hour**(*2*) / **Driver_state**(*OK*)
21	9	1.531	**Vehicle_characteristics**(*Mimo_CR*) >=< **Hour**(*2*)
22	23	1.502	**Vehicle_characteristics**(*Mimo_CR*) >=< **Hour**(*3*) / **Driver_state**(*OK*)
23	10	1.067	**Vehicle_characteristics**(*Mimo_CR*) >=< **Hour**(*3*)
24	24	1.423	**Vehicle_characteristics**(*Mimo_CR*) >=< **Hour**(*4*) / **Driver_state**(*OK*)
25	11	1.109	**Vehicle_characteristics**(*Mimo_CR*) >=< **Hour**(*4*)
26	25	0.300	**Vehicle_characteristics**(*Mimo_CR*) >=< **Hour**(*5*) / **Driver_state**(*OK*)
27	26	0.523	**Vehicle_characteristics**(*Mimo_CR*) >=< **Hour**(*22*) / **Driver_state**(*OK*)
28	12	0.375	**Vehicle_characteristics**(*Mimo_CR*) >=< **Hour**(*22*)
29	27	0.830	**Vehicle_characteristics**(*Mimo_CR*) >=< **Hour**(*23*) / **Driver_state**(*OK*)
30	13	0.579	**Vehicle_characteristics**(*Mimo_CR*) >=< **Hour**(*23*)
31	14	0.318	**Vehicle_characteristics**(*PCR*) >=< **Hour**(*13*)
32	28	0.366	**Vehicle_characteristics**(*PCR*) >=< **Day_of_Week**(*Sat*) / **Driver_state**(*OK*)

a) Hypotézy získané použitím kvantifikátoru AAD.

Nr.	Id	AvgDf	Hypothesis
1	7	-0.321	**Vehicle_characteristics**(*Soukr_nevydel*) >=< **Hour**(*8*) / **Driver_state**(*Vliv_alkoholu*)
2	4	-0.310	**Vehicle_characteristics**(*Soukr_org_podnik*) >=< **Day_of_Week**(*Sat*) / **Driver_state**(*OK*)
3	1	-0.328	**Vehicle_characteristics**(*Soukr_org_podnik*) >=< **Day_of_Week**(*Sat*)
4	5	-0.369	**Vehicle_characteristics**(*Soukr_org_podnik*) >=< **Day_of_Week**(*Sun*) / **Driver_state**(*OK*)
5	2	-0.381	**Vehicle_characteristics**(*Soukr_org_podnik*) >=< **Day_of_Week**(*Sun*)
6	3	-0.380	**Vehicle_characteristics**(*Mimo_CR*) >=< **Hour**(*7*)
7	6	-0.397	**Vehicle_characteristics**(*Mimo_CR*) >=< **Hour**(*7*) / **Driver_state**(*OK*)

b) Hypotézy získané použitím kvantifikátoru BAD.

Obrázek 5.5: Výsledné hypotézy pro úlohu Charakteristika vozidla ⇒ Čas nehody.

Nečekaná je hypotéza č. 5. Podle této hypotézy se za stejné podmínky, ale v otázce dne týdne, nehody služebních aut stávají častěji ve čtvrtek. Pro tuto hypotézu nejsem schopen vymyslet žádné smysluplné odůvodnění a proto ji beru pouze jako fakt, nikoliv jako materiál pro jakékoliv závěry.

Velmi zajímavá je hypotéza č. 6 (dohromady s hypotézou č. 7, která je však zanedbatelná kvůli opakování se s předchozí). Podle ní se nehody vozidel veřejné hromadné dopravy konají nejčastěji v pondělí. Je to snad nejpřekvapivější hypotéza nalezená v této úloze. Je ve své podstatě banální, ale zároveň nenaskytuje žádné smysluplné zdůvodnění. Rozdíl oproti průměru je velký: jedná se až o 22% podíl oproti 15% podílu v případě všech pondělních nehod. I v tom případě neumím říct, čím je tato skutečnost způsobena. Je však vidět, že i v tom, že obecně nemají lidé rádi pondělí, se něco skrývá.

Hypotézy č. 8–15 se týkají nehod vozidel státních společností a organizací. Nemá smysl přihlížet k podmínce: dobrý stav řidiče v tomto případě není příliš relevantní, jelikož tvoří 92 % všech záznamů s vyplněnou hodnotou tohoto atributu. Je však vidět, že se tyto nehody stávají nejčastěji v ranních hodinách. Nebyl zjištěn významný vliv alkoholu, přesto lze zjistit jednu zajímavou skutečnost. Tato kategorie vozidel výrazně ovlivňuje relativní počet nehod až do 10. hodiny (oproti služebním vozidlům nestátních firem a organizací), přičemž se to nejvíc projevuje po šesté hodině (dříve než u soukromých firem) a po deváté hodině (později). Může to vypovídat o tom, v jakých hodinách jezdí do práce služebními auty státní zaměstnanci. Je pozoruhodné, jak se takové skutečnosti projevují v rámci databáze nehod.

Hypotézy č. 16–30 se týkají vozidel registrovaných mimo ČR. Je zajímavé, že podle těchto hypotéz v nočních hodinách (22–5) havarují tato auta častěji. Možným zdůvodněním je to, že auta registrovaná v cizině jsou obvykle auta jedoucí na dalekou cestu nebo kamiony, kterých se tehdy na silnicích vyskytuje relativně mnohem více – Češi nemají potřebu nikam v tuto dobu jezdit, naopak dálková nákladní doprava s výhodou využívá prázdných silnic. Bylo by možné prozkoumat skladbu vozidel splňujících podmínky této hypotézy z hlediska typu vozidla a tím toto zdůvodnění potvrdit nebo zavrhnout.

Poslední dvě hypotézy (č. 31–32) podle mého názoru nedávají smysl. Nedá se však popřít, že se v datech vyskytuje jejich potvrzení – není však možné vysvětlit, proč policejní auta by měla havarovat zrovna ve 13 hodin nebo v sobotu. Je to důkaz, že lze někdy najít v datech vztahy, které jsou nesmyslné, a je potřeba je vždy odstranit ze souboru konečných řešení.

Zaměříme se ještě na hypotézy vygenerované opačnou úlohou, tj. použitím kvantifikátoru BAD (obr. 5.5b). Hned první hypotéza je velmi překvapivá – říká totiž, že pokud vezmeme v potaz nehody, kde byl řidič pod vlivem alkoholu, tak soukromá nevýdělečná vozidla nejméně havarují v 8 ráno. Znamená to nejspíš, že osoby dojíždějící ráno do práce volí jiný způsob dopravy, pokud jsou ještě pod vlivem alkoholu, oproti osobám řídícím služební automobily (viz 5.1), které nemají na vybranou. Zdá se tedy, že lidé jsou v této otázce zodpovědnější, než se běžně předpokládá – koneckonců se jedná o jejich soukromý majetek.

Podle hypotéz č. 2–5 jsou nehody služebních vozidel o víkendů nejméně časté. Je to přirozená skutečnost: víkend je čas odpočinku, proto se většinou služební auta nepoužívají.

Zajímavé je, že podle posledních dvou hypotéz (6–7) vozidla registrovaná mimo ČR nejméně havarují po sedmé hodině ranní. Může to být způsobeno buď velkým počtem nehod jiných vozidel, nebo tím, že si řidiči na dálkových trasách uvědomují, že v tuto dobu je na silnicích největší intenzita vozidel, kterými řidiči jezdí do práce.

5.3 Asociační pravidla III – složitější pravidla

Po dvou příkladech základních typů úloh procedury 4ft-Miner uvádím zde ještě jeden příklad složitějších pravidel. Na tomto příkladu uvedu některá pokročilejší nastavení cedentů, kterých lze využít ke generování méně obecných typů hypotéz. Analytická otázka zní:

> *Existuje nějaká kombinace specifického místa nehody (např. lesní parkoviště) a času, ve kterém se stala, pro kterou je některá příčina nehody nebo stav řidiče v jejím okamžiku výrazně častější?*

Pro tuto otázku nebyly stanoveny žádné předem očekávané výsledky.

Použitá nastavení

Úloha byla spuštěna s těmito parametry:

- procedura: 4ft-Miner,
- antecedent: min. délka 2, max. délka 4, dva parciální cedenty, první délky 1 s atributem *Specific_location*, druhý s min. délkou 1, max. délkou 3 a s atributy *Hour*, *Day_of_Week* a *Month* (všechny parametry s povolenou délkou 1, atributy *Specific_location* a *Hour* nastaveny jako důležité, ostatní pak jako vedlejší)
- sukcedent: min. délka 1, max. délka 2, dva atributy *Accident_cause* a *Driver_state* s maximálním povoleným počtem kategorií rovným 1,
- podmínka: prázdná,
- kvantifikátory: BASE 30, AAD 0,5.

Tato specifická nastavení antecedentu mají několik účelů, popíšu je podle jednotlivých parametrů. Prvním je použití dvou parciálních cedentů. V rámci každého parciálního cendentu lze samostatně nastavit jeho minimální i maximální délku a také logickou spojku používanou ke generování hypotéz (konjunkce nebo disjunkce). Rozdělením cedentu na více parciálních cedentů můžeme měnit logickou spojku v rámci části cedentu a také např. zajistit přítomnost daného literálu v pravidle vždy (oproti jiným literálům, které se v něm během generování vyskytovat nemusí), nebo dokonce rozdělit cedent na více skupin, z kterých vždy bude najednou vybrán zadaný počet atributů. Mezi jednotlivými parciálními cedenty se vždy aplikována logická spojka konjunkce. Je také potřeba brát v potaz, že celková délka cedentu (čili počet všech použitých literálů) je nastavována zvlášť (viz také 4.6.2).

Druhým využitým nastavením je označení daných atributů za důležité („basic“, *B*) a jiných za vedlejší („remaining“, *R*). Vygenerované budou vždy jen hypotézy, ve kterých se vyskytne alespoň jeden důležitý atribut **z každého dílčího cedentu, ve kterém je nějaký důležitý atribut definován** (viz také 1.2.3).

V případě této úlohy výsledkem je to, že se v každé hypotéze vždy vyskytnou literály tvořené atributy *Specific_location* a *Hour*, zatímco literály tvořené atributy *Day_of_Week* a *Month* se vyskytnout nemusí. Mnohdy lze dosáhnout stejných výsledků více způsoby. Tentokrát by to mohlo být zahrnutí prvních dvou atributů v rámci jednoho parciálního cedentu pevné délky 2, dvou posledních naopak v rámci druhého parciálního cedentu s min. délkou rovnou 0 a max. délkou rovnou 2.

Výsledky úlohy

Výsledky úlohy tvoří 99 hypotéz. Hypotézy mají tvar

$$\text{AvgDf Specific\_location}(X) \ \&\ \text{Hour}(Y)\ (\&\ \text{Day\_of\_Week}(Z)) >\div< \text{Accident\_cause}(W)\ (\&\ \text{Driver\_state}(V)),$$

kde X je charakteristika vozidla, Y je hodina konání nehody, Z je den v týdnu, W je příčina nehody, V je stav řidiče a *AvgDf* je číselný parametr (pro popis viz 5.1). Neuvádím zde seznam všech hypotéz z důvodu jeho většího objemu (99 hypotéz). Výsledky však lze rozdělit do několika skupin, které tvoří „shluky“. Z těchto „shluků“ vyberu hypotézy, které je dobře reprezentují, a ty okomentuji.

První skupinou jsou hypotézy typu

$$\text{Přechod pro chodce} \Rightarrow \text{Nedání přednosti chodci na přechodu.}$$

Je zřejmé, že tento typ hypotézy je tautologický: nedat přednost na přechodu lze jenom na přechodu. Odpovídá tomu parametr *AvgDf*, který dosahuje až hodnot rovných 60. Zajímavé však je, že nalezené hypotézy pokrývají pouze šestou a sedmou hodinu ranní a pak 13. až 18. hodinu. Je to nejspíše způsobeno kvantifikátorem BASE 30, který odstranil z výsledků hypotézy stávající se v ostatních hodinách. Plyne z toho, že mezi 8. a 13. hodinou se těchto nehod stává znatelně méně.

Podobná je skupina hypotéz typu

$$\text{Přechod pro chodce} + \text{Hodina} \Rightarrow \text{Nedodržení bezpečné vzdálenosti za vozidlem.}$$

Můžeme očekávat, že se v takovém případě jedná o najetí do vozidla, které prudce zabrzdilo před přechodem pro chodce. Spolu s hypotézami typu

$$\text{Blízko přechodu pro chodce} + \text{Hodina} \Rightarrow \text{Nedodržení bezpečné vzdálenosti za vozidlem}$$

tvoří velkou skupinu, ve které je suma hodnot *AvgDf* rovna přibližně 5. Hypotézy jsou rozděleny do dvou skupin pravděpodobně v následku nejasnosti, zda se nehoda stala už na přechodu, nebo ještě v jeho blízkosti, může to být také způsobeno brzděním před křižovatkami, blízko nichž se nachází přechod pro chodce. Tyto hypotézy pokrývají hodiny 6–17, tedy naprostou většinu dne. Nehody tohoto typu v ostatních hodinách jsou nejspíše odfiltrovány kvantifikátorem BASE 30.

Poslední velkou skupinou jsou hypotézy typu

$$\text{Parkoviště přiléhající ke komunikaci} + \text{Hodina} \Rightarrow \text{Nesprávné otáčení nebo couvání}$$

spolu s hypotézami typu

$$\text{Parkoviště přiléhající ke komunikaci} + \text{Hodina} \Rightarrow \text{Nevěnování se jízdě.}$$

Tyto hypotézy pokrývají hodiny 7–20, přičemž hypotézy prvního typu mají hodnotu *AvgDf* rovnou průměrně 5, zatímco hypotézy druhého typu mají tuto hodnotu rovnou průměrně 0,7. Ve většině případů je tedy u nehod tohoto typu viníkem řidič vyjíždějící z parkoviště.

Je vidět, že v databázi nehod lze najít i složitější pravidla, ze kterých můžeme vyčíst podrobnější informace. Čím podrobněji nastavíme úlohu, tím zajímavější hypotézy můžeme vygenerovat, má to však tu nevýhodu, že častěji se stane, že nenajdeme vůbec nic relevantního. Platí však, že čím podrobnější hypotéza je, tím menší je šance, že napadne někoho bez pomoci dataminingového softwaru. Stojí proto za to věnovat se takovémuto výzkumu.

Na obrázku 5.6 se nachází výňatek ze seznamu výsledků zobrazující předposlední skupinu hypotéz. Je zda také vidět nezohledněný (pro nevypovídající charakter výsledků) v komentářích atribut *Driver_state*.

Nr.	Id	AvgDf	Hypothesis
4	2	62.430	**Specific_location**(*Přechod_chodce*) & **Hour**(*6*) >=< **Accident_cause**(*Přechod_chodci_přechod*) & **Driver_state**(*OK*)
5	1	57.309	**Specific_location**(*Přechod_chodce*) & **Hour**(*6*) >=< **Accident_cause**(*Přechod_chodci_přechod*)
6	4	48.676	**Specific_location**(*Přechod_chodce*) & **Hour**(*7*) >=< **Accident_cause**(*Přechod_chodci_přechod*) & **Driver_state**(*OK*)
7	6	0.555	**Specific_location**(*Přechod_chodce*) & **Hour**(*7*) >=< **Accident_cause**(*Jízda_vzdálenost_za*) & **Driver_state**(*OK*)
8	3	48.046	**Specific_location**(*Přechod_chodce*) & **Hour**(*7*) >=< **Accident_cause**(*Přechod_chodci_přechod*)
9	5	0.564	**Specific_location**(*Přechod_chodce*) & **Hour**(*7*) >=< **Accident_cause**(*Jízda_vzdálenost_za*)
10	8	1.726	**Specific_location**(*Přechod_chodce*) & **Hour**(*13*) >=< **Accident_cause**(*Jízda_vzdálenost_za*) & **Driver_state**(*OK*)
11	7	1.643	**Specific_location**(*Přechod_chodce*) & **Hour**(*13*) >=< **Accident_cause**(*Jízda_vzdálenost_za*)
12	10	41.575	**Specific_location**(*Přechod_chodce*) & **Hour**(*14*) >=< **Accident_cause**(*Přechod_chodci_přechod*) & **Driver_state**(*OK*)
13	12	1.228	**Specific_location**(*Přechod_chodce*) & **Hour**(*14*) >=< **Accident_cause**(*Jízda_vzdálenost_za*) & **Driver_state**(*OK*)
14	9	37.914	**Specific_location**(*Přechod_chodce*) & **Hour**(*14*) >=< **Accident_cause**(*Přechod_chodci_přechod*)
15	11	1.152	**Specific_location**(*Přechod_chodce*) & **Hour**(*14*) >=< **Accident_cause**(*Jízda_vzdálenost_za*)
16	14	32.580	**Specific_location**(*Přechod_chodce*) & **Hour**(*15*) >=< **Accident_cause**(*Přechod_chodci_přechod*) & **Driver_state**(*OK*)
17	16	1.253	**Specific_location**(*Přechod_chodce*) & **Hour**(*15*) >=< **Accident_cause**(*Jízda_vzdálenost_za*) & **Driver_state**(*OK*)
18	13	30.812	**Specific_location**(*Přechod_chodce*) & **Hour**(*15*) >=< **Accident_cause**(*Přechod_chodci_přechod*)
19	15	1.172	**Specific_location**(*Přechod_chodce*) & **Hour**(*15*) >=< **Accident_cause**(*Jízda_vzdálenost_za*)
20	18	35.639	**Specific_location**(*Přechod_chodce*) & **Hour**(*16*) >=< **Accident_cause**(*Přechod_chodci_přechod*) & **Driver_state**(*OK*)
21	20	1.128	**Specific_location**(*Přechod_chodce*) & **Hour**(*16*) >=< **Accident_cause**(*Jízda_vzdálenost_za*) & **Driver_state**(*OK*)
22	17	35.065	**Specific_location**(*Přechod_chodce*) & **Hour**(*16*) >=< **Accident_cause**(*Přechod_chodci_přechod*)
23	19	1.172	**Specific_location**(*Přechod_chodce*) & **Hour**(*16*) >=< **Accident_cause**(*Jízda_vzdálenost_za*)
24	22	49.234	**Specific_location**(*Přechod_chodce*) & **Hour**(*17*) >=< **Accident_cause**(*Přechod_chodci_přechod*) & **Driver_state**(*OK*)
25	21	49.524	**Specific_location**(*Přechod_chodce*) & **Hour**(*17*) >=< **Accident_cause**(*Přechod_chodci_přechod*)
26	23	35.433	**Specific_location**(*Přechod_chodce*) & **Hour**(*18*) >=< **Accident_cause**(*Přechod_chodci_přechod*)
27	25	2.868	**Specific_location**(*Blízko_přechodu*) & **Hour**(*7*) >=< **Accident_cause**(*Jízda_vzdálenost_za*) & **Driver_state**(*OK*)
28	24	2.808	**Specific_location**(*Blízko_přechodu*) & **Hour**(*7*) >=< **Accident_cause**(*Jízda_vzdálenost_za*)
29	27	1.845	**Specific_location**(*Blízko_přechodu*) & **Hour**(*8*) >=< **Accident_cause**(*Jízda_vzdálenost_za*) & **Driver_state**(*OK*)
30	26	1.933	**Specific_location**(*Blízko_přechodu*) & **Hour**(*8*) >=< **Accident_cause**(*Jízda_vzdálenost_za*)

Obrázek 5.6: Výňatek ze seznamu výsledných hypotéz u složitější úlohy.

5.4 Frekvenční analýza I – monotónní posloupnosti

Charakter výsledků použití jednotlivých kvantifikátorů se v případě procedury CF-Miner liší poněkud více než u asociačních pravidel hledaných procedurou 4ft-Miner. Z toho důvodu je vhodné už v úvodu do popisu každé úlohy tohoto typu zdůraznit charakter histogramu, jaký nás zajímá. Analytická otázka pro první úlohu zní:

> *Existují nějaké parametry nehody, pro které platí, že je posloupnost frekvencí takových nehod v jednotlivých letech (přibližně) klesající nebo rostoucí?*

Pro tuto otázku nebyly stanoveny žádné předem očekávané výsledky.

Použitá nastavení

Úloha byla spuštěna s těmito parametry:

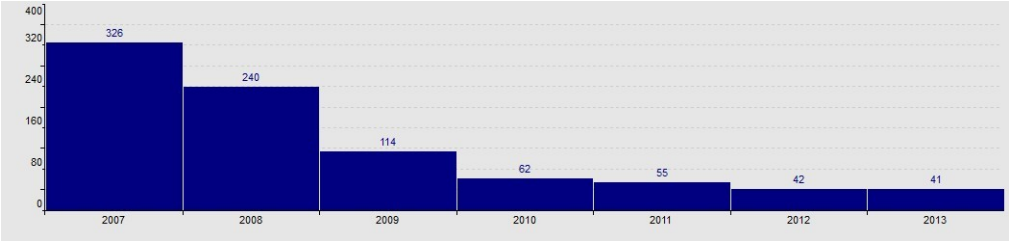
- procedura: CF-Miner,
- atribut pro tvorbu histogramu: *Year*,
- podmínka: délka 1, vybrány atributy: *Driver_state*, *Responsibility*, *Accident_cause* a *Road_number* s maximálním povoleným počtem kategorií rovným 1,
- kvantifikátory: SUM 20, S-DN $\geq 70\%$ (popř. S-UP $\geq 70\%$).

Kvantifikátor SUM zajišťuje, že jsou brány v potaz pouze takové typy nehod, jejichž celkový počet za sledované období byl roven alespoň 20. Kvantifikátory S-DN a S-UP hledají takové parametry, při nichž je počet „schodů“ v příslušném směr roven alespoň 70 % počtu kategorií (v tomto případě 6). U kvantifikátoru S-DN byla navíc aplikovaná možnost hledání pouze takových parametrů, pro které tvoří tyto schody jednu posloupnost, nikoliv např. dvě rozdělené posloupnosti. Pro atribut *Road_number* byl použit pouze kvantifikátor S-UP, pro ostatní atributy byly použity oba kvantifikátory.

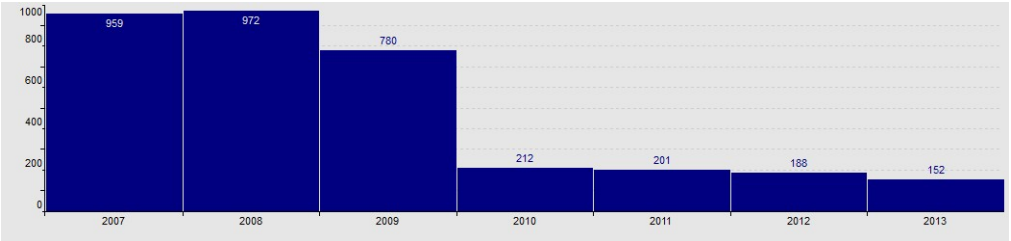
Výsledky úloh

Jelikož reálnou náplní každé hypotézy je histogram, nemá smysl zveřejňovat zde seznamy všech hypotéz pro jednotlivá spuštění úloh – nic by z nich neplynulo a neměly by význam. Na druhou stranu zveřejnit zde všechny histogramy je taktéž nemožné kvůli nedostatku místa. Vybral jsem proto několik zajímavých histogramů, které jsem doplnil komentářem.

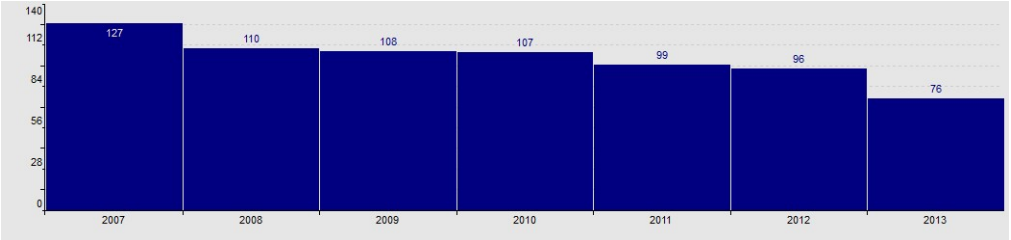
Na obr. 5.7a vidíme histogram nehod, jejichž příčinou je nedání přednosti při přejíždění z pruhu do pruhu. Je zřejmé, že počet těchto nehod stále klesal až do velmi nízkých hodnot v roce 2013. Podobně, což je trochu překvapující, se má situace v otázce nehod, při kterých byl řidič pod vlivem alkoholu (obr. 5.7b). Počet těchto nehod razantně klesl mezi lety 2009 a 2010 a od té doby pořád klesá. Tak razantní pokles je nejspíše způsoben zvýšením hodnoty finanční škody, od které je nutno nahlásit nehodu PČR. Osoby pod vlivem alkoholu samozřejmě nemají zájem o policejní kontrolu a pokud je to jen možné, nehodu „utají“. Pokles (i když



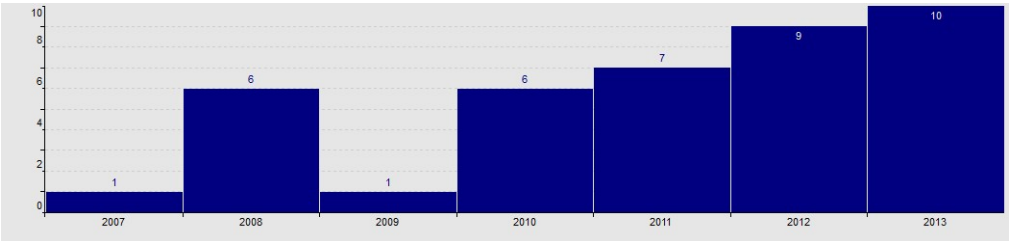
a) Nehody, jejichž příčinou bylo nedání přednosti při přejíždění z pruhu do pruhu.



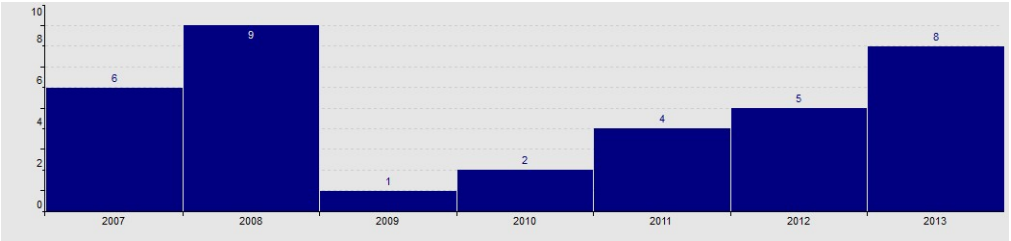
b) Nehody, při kterých byl řidič pod vlivem alkoholu.



c) Nehody, jejichž viníkem byl chodec.



d) Nehody, při kterých byl řidič nemocný, po úrazu apod.



e) Nehody na silnici III/2411.

Obrázek 5.7: Histogramy vybraných typů nehod v jednotlivých létech.

mnohem méně výrazný) je vidět také na obr. 5.7c v případě nehod, ve kterých zodpovědnost nesl chodec.

Naopak mírný růst je vidět na obr. 5.7d, na kterém je znázorněn histogram nehod, při kterých byl řidič nemocný, po úrazu či měl jiné zdravotní problémy. Nepočítaje roku 2009, můžeme vidět stálý růst naštěstí jinak malého počtu těchto nehod. Zajímavý je taky histogram na obr. 5.7e, který znázorňuje nehody na silnici III/2411. Je vidět růst v prvních dvou letech, silný pokles v roku 2009 a následně opět růst do původních hodnot. Tato silnice (spolu se silnicí III/1214, pro kterou je histogram téměř stejný, není však umístěn v této práci) by si zasloužila zvýšenou pozornost příslušných složek, jelikož se jedná o jediné dvě silnice ve Středočeském kraji, na kterých je růst tak výrazný.

Výše uvedené příklady jasně ukazují, že pomocí kvantifikátorů S-DN a S-UP lze v datech vyhledat zajímavé časové řady. Pokud by byla databáze větší a pokrývala např. 20 let, byly by výsledky ještě zajímavější. I při tomto kratším období sledování jsou však výsledky pozoruhodné.

5.5 Frekvenční analýza II – malý rozptyl hodnot

Pokud se podíváme na histogram nehod v závislosti na dnu v týdnu (obr. 5.8), zjistíme známou věc, totiž že počet nehod je přibližně stejný od pondělí do čtvrtka, v pátek je mírně větší, o víkendu pak (zvláště v neděli) je nejmenší. Zajímá nás, zda existují nějaké nehody, u kterých tento trend neplatí, přesněji takové, pro které počet nehod ve všechny dny v týdnu je přibližně stejný. Analytická otázka v tomto případě zní:

> *Existují nějaké parametry nehody, pro které platí, že jsou jejich frekvence ve všechny dny v týdnu přibližně stejné?*

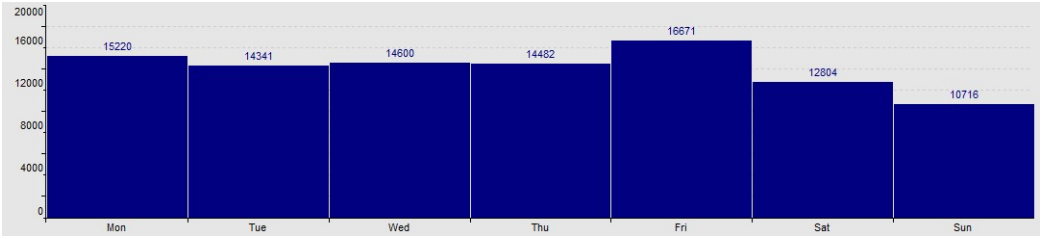
Pro tuto otázku nebyly stanoveny žádné předem očekávané výsledky.

Použitá nastavení

Úloha byla spuštěna s těmito parametry:

- procedura: CF-Miner,
- atribut pro tvorbu histogramu: *Day_ of_ Week*,
- podmínka: délka 1, vybrány atributy mj. *Accident_ localization*, *Layout*, *Surface_ type*, *Vehicle_ type* a *Solid_ object_ type* s maximálním počtem kategorií rovným 1,
- kvantifikátory: SUM 20, MIN $\geq$ 70%.

Kvantifikátor SUM zajišťuje, že jsou brány v potaz pouze takové typy nehod, jejichž celkový počet za sledované období byl roven alespoň 20. Kvantifikátor MIN hledá takové histogramy, pro které nejmenší frekvence je rovna alespoň 70 % nejvyšší frekvence, tedy ve kterých se hodnoty všech frekvencí nachází v mnohem menším intervalu.



Obrázek 5.8: Histogram nehod v jednotlivých dnech v týdnu.

Výsledky úloh

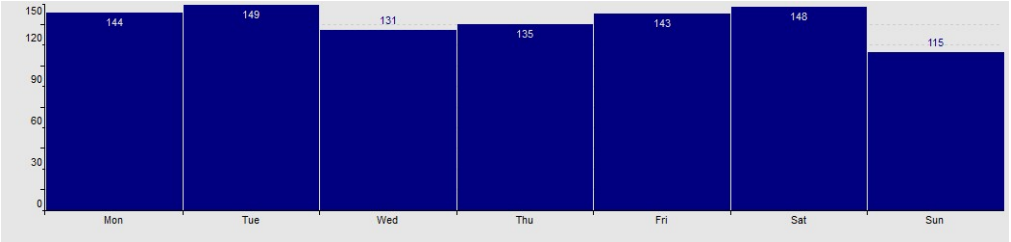
Výsledné histogramy jsou v některých případech překvapivé. Na obr. 5.9a a 5.9b se nachází histogramy nehod na chodnících či ostrůvcích o nehod na cestách se štěrkovým povrchem. Překvapivé je už to, že jejich počty jsou téměř shodné, na čemž nejzajímavější je shodná podoba histogramu o víkendu. Mírný narůst v sobotu oproti pátku a pokles v neděli může být způsoben např. větší tendencí vyrážet na výlety v sobotu, což se projevuje navštěvováním štěrkových silnic a větším počtem méně pozorných chodců na ostrůvcích. Taková argumentace není však ničím dodatečným podložena a je spíše pokusem o vysvětlení než reálným zdůvodněním.

Na obr. 5.9c se nachází histogram nehod osobních automobilů. Je vidět výrazně větší počet nehod v pátek – rozdíl mezi čtvrtkem a pátkem téměř odpovídá rozdílu, který je mezi těmito dny vidět na histogramu všech nehod na obr. 5.8. V sobotu však počet nehod klesá na původní hodnotu, nikoliv na nižší, pokles v neděli je také mírnější než na histogramu všech nehod. Znamená to, že zvýšený počet nehod v pátek je z větší části způsoben osobními automobily, kterých se v pátek na silnicích vyskytuje mnohem více kvůli konci pracovního týdne, avšak víkendový pokles je způsoben jinou skupinou automobilů (nejspíše autobusů, kamionů atd.).

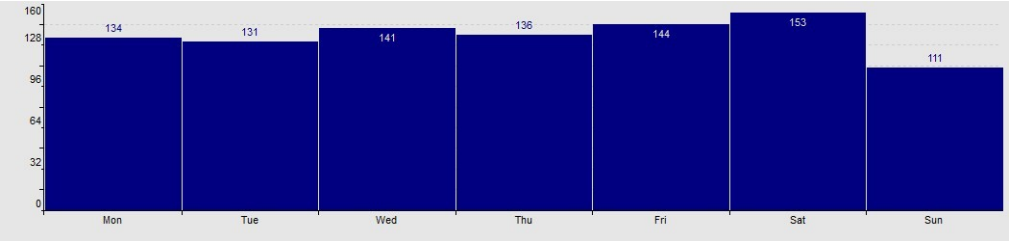
Na obr. 5.9d jsou znázorněny počty nehod, které se odehrály v oblouku. Zde je rozdíl mezi nedělním počtem a počty v pracovních dnech velmi malý, v pátek a v sobotu naopak je růst velký. Pokud vezmeme v potaz závěry z předchozího odstavce, nebude příliš odvážné říci, že jsou víkendové nehody v oblouku zaviněny mnohem spíš osobními automobily než kamiony.

Na posledním obrázku (5.9e) je umístěn histogram srážek se sloupem. Je překvapivé, že se největší počet těchto srážek koná v pondělí. Hlavním poznatkem je však to, že kromě pondělí je počet nehod přibližně stejný ve všechny dny v týdnu, tedy i o víkendu.

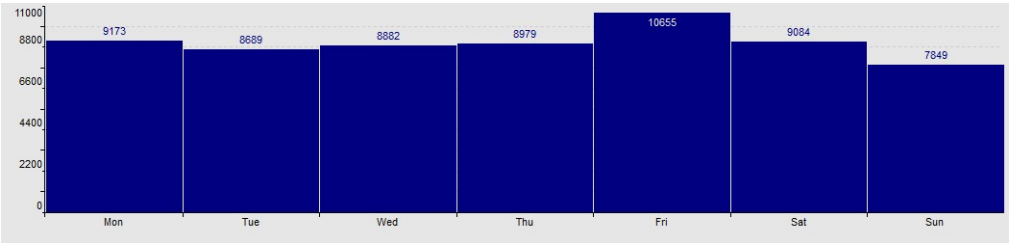
Na základě výše uvedených úvah lze říci, že i přes výrazné rozdíly v celkovém počtu nehod ne všechny jejich typy jsou závislé na dnu v týdnu. Lze připustit závěr, že existují v tomto souboru dat jevy čistě náhodné (náhodou je srážka zrovna se sloupem místo např. se stromem), dále také, že typické mínění o tom, proč je v daný den v týdnu nehod méně či více, není úplně pravdivé (protože suchá data tvrdí něco jiného). Tohoto typu zkoumání je jistě přínosné a může někdy přinést nečekané výsledky.



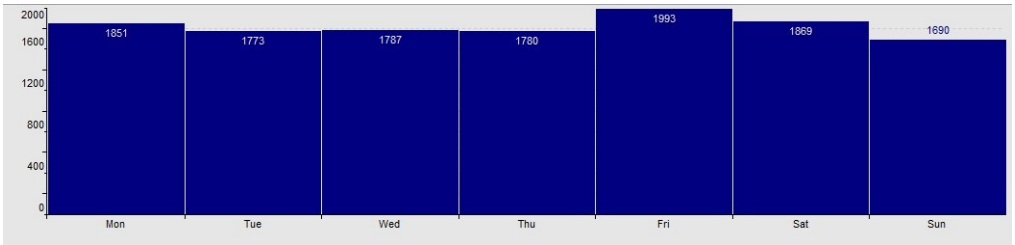
a) Nehody na chodníku či ostrůvku.



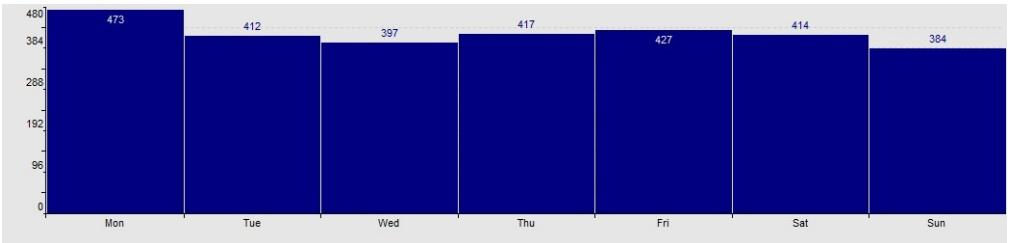
b) Nehody na cestách se štěrkovým povrchem.



c) Nehody osobních automobilů.



d) Nehody v oblouku.



e) Srážky se sloupy.

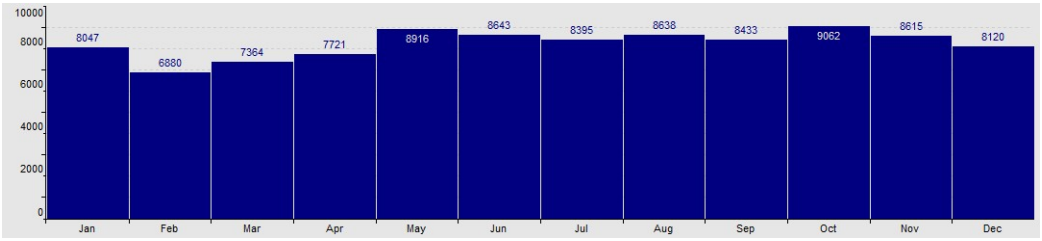
Obrázek 5.9: Histogramy vybraných typů nehod v jednotlivých dnech v týdnu.

5.6 Frekvenční analýza III – velký rozptyl hodnot

Tentokrát základem pro úlohu je histogram nehod založený na měsících (obr. 5.10). Je vidět, že rozložení nehod je v jednotlivých měsících přibližně stejné – hlavní výjimku tvoří únor, který je však kratší než ostatní měsíce, což má na počet nehod určitý vliv. Zajímá nás proto, zda existují nějaké nehody, u kterých má měsíc viditelný vliv na jejich počet. Analytická otázka pro tuto úlohu zní:

> *Existují nějaké parametry nehody, pro které platí, že jejich frekvence v jednotlivých měsících nejsou stejné?*

Pro tuto otázku nebyly stanoveny žádné předem očekávané výsledky.



Obrázek 5.10: Histogram nehod pro jednotlivé měsíce.

Použitá nastavení

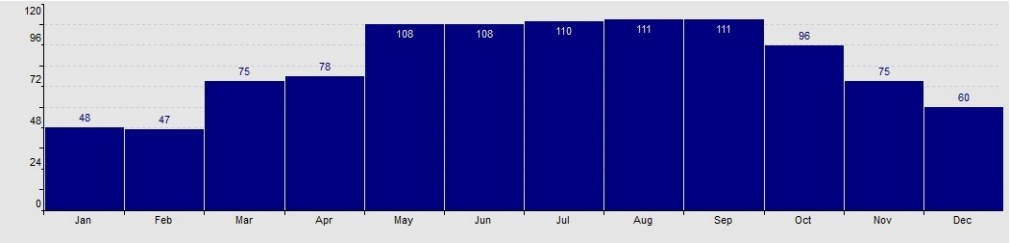
Úloha byla spuštěna s těmito parametry:

- procedura: CF-Miner,
- atribut pro tvorbu histogramu: *Month*,
- podmínka: délka 1, vybrány atributy mj. *Driver_state*, *Vehicle_object_characteristics*, *Vehicle_type* a *Visibility* s maximálním povoleným počtem kategorií rovným 1,
- kvantifikátory: SUM 20, MIN $\leq$ 50%.

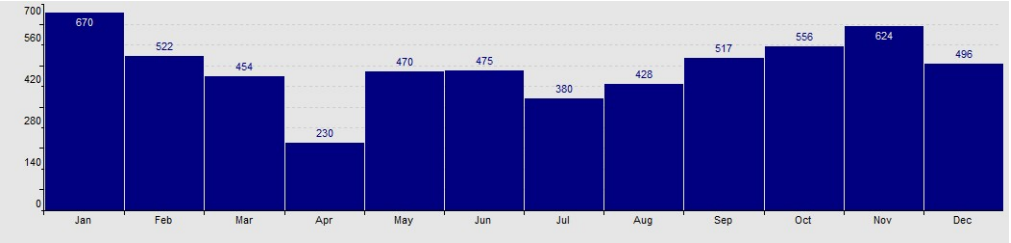
Kvantifikátor SUM zajišťuje, že jsou brány v potaz pouze takové typy nehod, jejichž celkový počet za sledované období byl roven alespoň 20. Kvantifikátor MIN tentokrát hledá takové histogramy, pro které je nejmenší frekvence rovna nejvýše 50 % největší frekvence, tedy ve kterých se hodnoty všech frekvencí nachází v delším intervalu.

Výsledky úloh

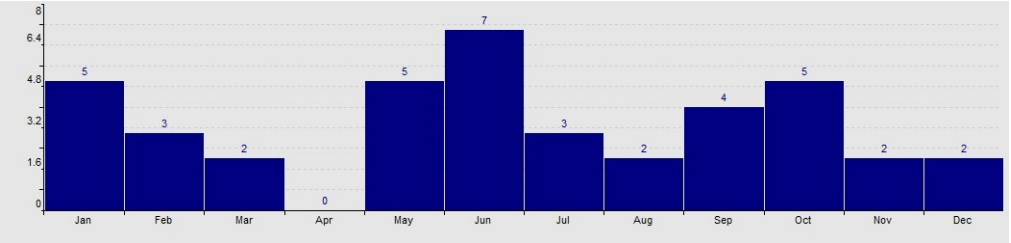
Na prvním histogramu (obr. 5.11a) vidíme nehody, při kterých řidič usnul nebo byl unaven. Přestože obecně není počet nehod v létě vyšší, jsou zde vidět výrazně vyšší počty nehod od května do září a naopak až dvakrát nižší počty těchto nehod v prosinci až únoru.



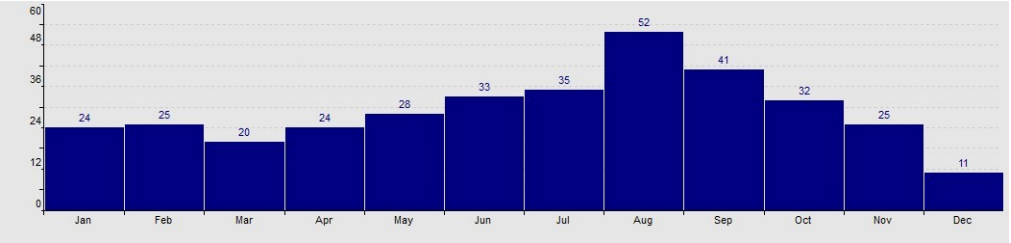
a) Nehody, při kterých byl řidič unaven nebo usnul.



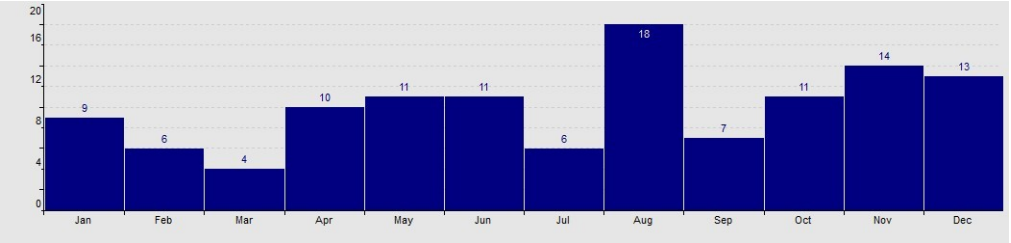
b) Nehody, při kterých byla viditelnost zhoršena povětrnostními podmínkami (během dne).



c) Nehody, při kterých byl řidič nemocný, po úrazu apod.



d) Nehody traktorů.



e) Nehody vozidel Ministerstva vnitra.

Obrázek 5.11: Histogramy vybraných typů nehod v jednotlivých měsících.

Důvodem této skutečnosti může být dlouhý den a tendence k delším jízdám bez zastavení či změny řidiče (např. návraty z dovolené v Chorvatsku a únava na koncovém úseku). Jiným zdůvodněním může být taky rostoucí teplota, která negativně ovlivňuje soustředění řidičů aut nevybavených klimatizací.

Na obr. 5.11b se nachází histogram nehod při viditelnosti zhoršené povětrnostními podmínkami (pouze během dne). Zde je naopak vidět, že počty těchto nehod jsou nejvyšší v lednu a v listopadu. V prvním případě se jedná nejspíše o silná sněžení, ve druhém o zvýšený výskyt mlhy. Zajímavý je velmi nízký počet nehod tohoto typu v dubnu (až dvakrát nižší než v sousedních měsících). Srovnání této skutečnosti s meteorologickými daty by mohlo přinést zajímavé závěry. Ještě zajímavější je v souvislosti s tím histogram na obr. 5.11c. Počet záznamů, na jejichž základě je tento histogram vytvořen, je velmi malý, i přesto je však pozoruhodné, že se žádná nehoda, při které byl řidič nemocný, nestala v dubnu.

Na obr. 5.11d jsou znázorněny počty nehod traktorů. Nejnižší počet těchto nehod se stává v prosinci, nejvyšší naopak v srpnu, v jiných měsících počty plynule rostou a klesají směrem k extrémním hodnotám. Nejvyšší počet nehod v srpnu je samozřejmostí – je to období žní, nadprůměrně vysoký počet těchto nehod v sousedních měsících je zdůvodněn podobně.

Obr. 5.11e je příkladem skutečnosti těžko vysvětlitelné až náhodné. Nehody vozidel Ministerstva vnitra neodpovídají žádným z ostatních rozdělení, jsou nejspíše náhodné, popř. spojené s nezjistitelným rozvrhem politických událostí. Ještě větším překvapením je velmi vysoký počet nehod v srpnu ve srovnání s velmi nízkými jejich počty v červenci a září. Možná jezdí zaměstnanci MV v srpnu na dovolenou a půjčují se na ni služební auto?

Tato úloha ukazuje, že lze v datech najít vztahy jasné a vysvětlitelné, ale také skutečnosti nepochopitelné a překvapivé. Data mining nabízí zkoumání závislostí obou typů. Badatel může toho s výhodou využít a potvrdit známé fakty, rozšířit své znalosti – nebo se někdy i pobavit, což při dlouhé a intenzivní práci není nepodstatné.

5.7 Využití geografických dat I – školy

Tato a následující sekce se věnují posouzení přínosu použití odvozených geografických dat v data miningu. Náplní první úlohy je hledání vztahu mezi vzdáleností od školy a typem vozidla, které danou nehodu způsobilo. Analytická otázka pro tuto úlohu zní:

> *Existuje nějaký typ vozidla, pro které je frekvence nehod závislá na vzdálenosti od školy?*

Pro tuto otázku nebyly stanoveny žádné předem očekávané výsledky.

Použitá nastavení

Úloha byla spuštěna s těmito parametry:

- procedura: 4ft-Miner,
- antecedent: délka 1, atribut *School_ distance* s maximálním počtem kategorií rovným 1 (standardní s použitím standarních koeficientů typu „subset“), nebo 15 (koeficienty typu „left cut“),

- sukcedent: délka 1, dva atributy *Vehicle_ type* s maximálním povoleným počtem kategorií rovným 1,
- podmínka: prázdná,
- kvantifikátory: BASE 20, AAD 0,3 (poté i CHI $\alpha = 0{,}01$).

To znamená, že jsou v datech hledány takové hypotézy, podle kterých na dané silnici je poměr nehod daného typu vozidla vůči všem nehodám na této silnici alespoň o 50 % větší, než je poměr všech nehod tohoto typu vozidla ke všem nehodám vůbec. Hypotéza se pokládá za relevantní, pokud je počet nehod daného typu vozidel na dané silnici roven alespoň 20.

Atribut *School_ distance* byl využit dvěma různými způsoby. První je standardní a odpovídá většině případů uvedených v předchozích úlohách. Druhý využívá koeficientů typu „left cut“, tedy „levý řez“. Tento typ funguje tak, že se vybírá daný souvislý soubor kategorií, který začíná vždy první kategorií. To umožňuje v případě této úlohy posuzovat všechny případy v okruhu o poloměru, který roste po každém kroku.

Kvantifikátor CHI je založen na χ^2 rozdělení a vybírá hypotézy, pro které je příslušná hodnota (stanovená výpočtem z čtyřpolní tabulky) větší než $1 - 2\alpha$ kvantil χ^2 rozdělení s jedním stupněm volnosti. Podrobnější popis najde čtenář v [8].

V první etapě byla úloha spuštěna s kvantifikátorem AAD 0,3 postupně s oběma typy koeficientů. V další etapě byla pro ještě lepší posouzení nezávislosti spuštěna s kvantifikátorem CHI $\alpha = 0{,}01$.

Výsledky úloh

Na obrázku 5.12 se nachází seznam výsledných hypotéz s jednou kategorií. Hypotézy jsou seřazeny podle sukcedentu, v další řadě pak podle antecedentu. Je vidět, že závislosti byly nalezeny pro několik typů vozidel. Pro různé typy vozidel byly závislosti odlišné.

Hypotézy č. 1–3 se týkají motocyklů. Je vidět, že s rostoucí vzdáleností od školy roste i relativní počet motocyklových nehod. Může jít o nehody způsobené nebezpečnou jízdou, jakou motocyklisté podstupují pro zábavu. 5–20 km od školy znamená většinou nezastavěnou oblast, která se k takové jízdě hodí mnohem více. Naopak blízko škol, tedy ve městech či vesnicích, se tyto nehody nekonají tak často.

Hypotézy č. 4–6 říkají, že se nehody automobilů s návěsy a přívěsy, tedy určených k dopravě nákladu, stávají nejčastěji (opět v přeneseném smyslu, viz poznámka pod čarou v sekci 5.2) ve velké vzdálenosti od škol, tedy také mimo zastavěnou oblast. Je to způsobeno pravděpodobně zvýšenou pozorností řidičů takových automobilů v zastavěné oblasti. Druhým důvodem je nejspíše mnohem větší počet nehod typických pro obec (špatné parkování, jízdní kola, pěší atd.), které podíl nehod nákladních automobilů zmenšují.

Na podobném principu jsou nejspíše založeny hypotézy týkající se autobusů, traktorů a jízdních kol (č. 7–10, 11–12, 13–16). Jedná se opět o nehody, které se podstatně častěji stávají ve městech (jízdní kolo), nebo naopak mimo obce (traktory).

Zajímavou skupinu tvoří hypotézy č. 17–24. Je vidět rostoucí relativní počet nehod, ve kterých zúčastněné vozidlo nebylo zjištěno, pokud vzdálenost od školy klesá. Může to znamenat, že řidiči, kteří způsobí škodu či nehodu poblíž školy – možná způsobenou nezodpovědným chováním dětí – častěji utíkají z místa nehody. Rostoucí tendence působí proti

tvrzení, že je to způsobeno charakterem nehod typickým pro město (jinými slovy: pokud mimo obec dojde k nehodě, většinou není kvůli jejímu charakteru možné, aby její pachatel utekl) – v tom případě by totiž v rámci kategorií např. do 1 000 m byla hodnota parametru *AvgDf* náhodná, nikoliv tvořící monotónní posloupnost.

Jelikož tvořily poslední hypotézy zajímavou skupinu, zopakoval jsem úlohu s použitím koeficientů typu „left cut“. Výsledky jsou znázorněny na obr. 5.13. Je vidět, že malá skupina hypotéz týkajících se nehod autobusů (č. 1–3) potvrzuje výsledky předchozí úlohy. Ještě výraznější je tento trend v případě nezjištěného typu vozidla: všechny hypotézy až do 1 000 m (č. 4–13) mají vysokou hodnotu *AvgDf*, která navíc klesá pomaleji než v případě předchozí úlohy. Je vidět, že je vskutku v datech zastoupen vztah nepřímé úměry mezi vzdáleností od školy a frekvencí nehod, ve kterých nebyl zjištěn typ vozidla.

Ve druhé etapě jsem pro ještě lepší posouzení toho, zda jsou tyto dva parametry vskutku závislé, zopakoval obě úlohy s použitím kvantifikátoru CHI $\alpha = 0{,}01$. Na obr. 5.14 a 5.15 se nachází seznamy výsledných hypotéz. V obou případech je vidět stejný vztah, dokonce v případě koeficientů „left cut“ rozšířený o více kategorií vzdálenosti a o jízdní kola. Pokles hodnoty *ChiSq* ve spojení se vzdáleností jasně znázorňuje vztah mezi těmito atributy.

Závěrem můžeme konstatovat, že vzdálenost od školy je parametr, jehož využití umožnilo nalezení vztahu s podílem nehod bez zjištěného typu vozidla. Je to první znamení, že data mining na datech odvozených pomocí geografických informačních systémů může generovat zajímavé výsledky.

Nr.	Id	AvgDf	Hypothesis
1	17	0.330	**School_distance**(*<5000;10000)*) >+< **Vehicle_type**(*Motocykl*)
2	19	0.373	**School_distance**(*<10000;15000)*) >+< **Vehicle_type**(*Motocykl*)
3	24	0.573	**School_distance**(*<15000;20000)*) >+< **Vehicle_type**(*Motocykl*)
4	20	0.436	**School_distance**(*<10000;15000)*) >+< **Vehicle_type**(*Osob_auto_prives*)
5	21	0.317	**School_distance**(*<10000;15000)*) >+< **Vehicle_type**(*Nakl_auto_prives*)
6	22	0.308	**School_distance**(*<10000;15000)*) >+< **Vehicle_type**(*Nakl_auto_naves*)
7	4	0.507	**School_distance**(*<200;300)*) >+< **Vehicle_type**(*Autobus*)
8	6	0.529	**School_distance**(*<300;400)*) >+< **Vehicle_type**(*Autobus*)
9	9	0.375	**School_distance**(*<500;600)*) >+< **Vehicle_type**(*Autobus*)
10	15	0.536	**School_distance**(*<800;900)*) >+< **Vehicle_type**(*Autobus*)
11	18	0.586	**School_distance**(*<5000;10000)*) >+< **Vehicle_type**(*Traktor*)
12	23	0.550	**School_distance**(*<10000;15000)*) >+< **Vehicle_type**(*Traktor*)
13	2	0.311	**School_distance**(*<100;200)*) >+< **Vehicle_type**(*Jizdni_kolo*)
14	10	0.330	**School_distance**(*<500;600)*) >+< **Vehicle_type**(*Jizdni_kolo*)
15	12	0.404	**School_distance**(*<600;700)*) >+< **Vehicle_type**(*Jizdni_kolo*)
16	13	0.497	**School_distance**(*<700;800)*) >+< **Vehicle_type**(*Jizdni_kolo*)
17	1	0.690	**School_distance**(*<100*) >+< **Vehicle_type**(*Nezjisteno*)
18	3	0.663	**School_distance**(*<100;200)*) >+< **Vehicle_type**(*Nezjisteno*)
19	5	0.555	**School_distance**(*<200;300)*) >+< **Vehicle_type**(*Nezjisteno*)
20	7	0.561	**School_distance**(*<300;400)*) >+< **Vehicle_type**(*Nezjisteno*)
21	8	0.492	**School_distance**(*<400;500)*) >+< **Vehicle_type**(*Nezjisteno*)
22	11	0.431	**School_distance**(*<500;600)*) >+< **Vehicle_type**(*Nezjisteno*)
23	14	0.323	**School_distance**(*<700;800)*) >+< **Vehicle_type**(*Nezjisteno*)
24	16	0.369	**School_distance**(*<800;900)*) >+< **Vehicle_type**(*Nezjisteno*)

Obrázek 5.12: Výsledné hypotézy pro vzdálenost od školy (AAD, subset).

Nr.	Id	AvgDf	Hypothesis
1	4	0.348	**School_distance**(<*400*) >=< **Vehicle_type**(*Autobus*)
2	6	0.333	**School_distance**(<*500*) >=< **Vehicle_type**(*Autobus*)
3	8	0.341	**School_distance**(<*600*) >=< **Vehicle_type**(*Autobus*)
4	1	0.690	**School_distance**(<*100*) >=< **Vehicle_type**(*Nezjisteno*)
5	2	0.670	**School_distance**(<*200*) >=< **Vehicle_type**(*Nezjisteno*)
6	3	0.618	**School_distance**(<*300*) >=< **Vehicle_type**(*Nezjisteno*)
7	5	0.601	**School_distance**(<*400*) >=< **Vehicle_type**(*Nezjisteno*)
8	7	0.577	**School_distance**(<*500*) >=< **Vehicle_type**(*Nezjisteno*)
9	9	0.550	**School_distance**(<*600*) >=< **Vehicle_type**(*Nezjisteno*)
10	10	0.507	**School_distance**(<*700*) >=< **Vehicle_type**(*Nezjisteno*)
11	11	0.486	**School_distance**(<*800*) >=< **Vehicle_type**(*Nezjisteno*)
12	12	0.475	**School_distance**(<*900*) >=< **Vehicle_type**(*Nezjisteno*)
13	13	0.448	**School_distance**(<*1000*) >=< **Vehicle_type**(*Nezjisteno*)

Obrázek 5.13: Výsledné hypotézy pro vzdálenost od školy (AAD, left cut).

Nr.	Id	Chi-Sq	Hypothesis
1	3	5.562	**School_distance**(<*300*) >=< **Vehicle_type**(*Autobus*)
2	5	13.584	**School_distance**(<*400*) >=< **Vehicle_type**(*Autobus*)
3	8	16.393	**School_distance**(<*500*) >=< **Vehicle_type**(*Autobus*)
4	11	21.766	**School_distance**(<*600*) >=< **Vehicle_type**(*Autobus*)
5	14	17.734	**School_distance**(<*700*) >=< **Vehicle_type**(*Autobus*)
6	17	17.617	**School_distance**(<*800*) >=< **Vehicle_type**(*Autobus*)
7	20	24.169	**School_distance**(<*900*) >=< **Vehicle_type**(*Autobus*)
8	23	20.223	**School_distance**(<*1000*) >=< **Vehicle_type**(*Autobus*)
9	26	16.165	**School_distance**(<*2000*) >=< **Vehicle_type**(*Autobus*)
10	29	12.790	**School_distance**(<*3000*) >=< **Vehicle_type**(*Autobus*)
11	31	5.489	**School_distance**(<*4000*) >=< **Vehicle_type**(*Autobus*)
12	6	4.808	**School_distance**(<*400*) >=< **Vehicle_type**(*Jizdni_kolo*)
13	9	6.382	**School_distance**(<*500*) >=< **Vehicle_type**(*Jizdni_kolo*)
14	12	11.310	**School_distance**(<*600*) >=< **Vehicle_type**(*Jizdni_kolo*)
15	15	17.508	**School_distance**(<*700*) >=< **Vehicle_type**(*Jizdni_kolo*)
16	18	26.457	**School_distance**(<*800*) >=< **Vehicle_type**(*Jizdni_kolo*)
17	21	29.447	**School_distance**(<*900*) >=< **Vehicle_type**(*Jizdni_kolo*)
18	24	26.743	**School_distance**(<*1000*) >=< **Vehicle_type**(*Jizdni_kolo*)
19	27	4.404	**School_distance**(<*2000*) >=< **Vehicle_type**(*Jizdni_kolo*)
20	1	69.313	**School_distance**(<*100*) >=< **Vehicle_type**(*Nezjisteno*)
21	2	253.102	**School_distance**(<*200*) >=< **Vehicle_type**(*Nezjisteno*)
22	4	407.806	**School_distance**(<*300*) >=< **Vehicle_type**(*Nezjisteno*)
23	7	568.745	**School_distance**(<*400*) >=< **Vehicle_type**(*Nezjisteno*)
24	10	692.951	**School_distance**(<*500*) >=< **Vehicle_type**(*Nezjisteno*)
25	13	798.172	**School_distance**(<*600*) >=< **Vehicle_type**(*Nezjisteno*)
26	16	805.519	**School_distance**(<*700*) >=< **Vehicle_type**(*Nezjisteno*)
27	19	860.004	**School_distance**(<*800*) >=< **Vehicle_type**(*Nezjisteno*)
28	22	931.526	**School_distance**(<*900*) >=< **Vehicle_type**(*Nezjisteno*)
29	25	924.305	**School_distance**(<*1000*) >=< **Vehicle_type**(*Nezjisteno*)
30	28	576.210	**School_distance**(<*2000*) >=< **Vehicle_type**(*Nezjisteno*)
31	30	398.151	**School_distance**(<*3000*) >=< **Vehicle_type**(*Nezjisteno*)
32	32	289.321	**School_distance**(<*4000*) >=< **Vehicle_type**(*Nezjisteno*)
33	33	183.510	**School_distance**(<*5000*) >=< **Vehicle_type**(*Nezjisteno*)
34	34	83.699	**School_distance**(<*10000*) >=< **Vehicle_type**(*Nezjisteno*)

Obrázek 5.14: Výsledné hypotézy pro vzdálenost od školy (CHI, left cut).

Nr.	Id	Chi-Sq	Hypothesis
1	30	63.431	**School_distance**(*<5000;10000)*) >+< **Vehicle_type**(*Motocykl*)
2	36	18.445	**School_distance**(*<10000;15000)*) >+< **Vehicle_type**(*Motocykl*)
3	41	6.668	**School_distance**(*<15000;20000)*) >+< **Vehicle_type**(*Motocykl*)
4	20	3.623	**School_distance**(*<1000;2000)*) >+< **Vehicle_type**(*Osob_auto*)
5	23	7.440	**School_distance**(*<2000;3000)*) >+< **Vehicle_type**(*Osob_auto*)
6	27	3.913	**School_distance**(*<4000;5000)*) >+< **Vehicle_type**(*Osob_auto*)
7	28	4.179	**School_distance**(*<4000;5000)*) >+< **Vehicle_type**(*Osob_auto_prives*)
8	31	4.870	**School_distance**(*<5000;10000)*) >+< **Vehicle_type**(*Osob_auto_prives*)
9	37	9.107	**School_distance**(*<10000;15000)*) >+< **Vehicle_type**(*Osob_auto_prives*)
10	18	3.534	**School_distance**(*<900;1000)*) >+< **Vehicle_type**(*Nakl_auto*)
11	21	4.318	**School_distance**(*<1000;2000)*) >+< **Vehicle_type**(*Nakl_auto*)
12	25	11.928	**School_distance**(*<3000;4000)*) >+< **Vehicle_type**(*Nakl_auto*)
13	32	4.737	**School_distance**(*<5000;10000)*) >+< **Vehicle_type**(*Nakl_auto_prives*)
14	38	8.745	**School_distance**(*<10000;15000)*) >+< **Vehicle_type**(*Nakl_auto_prives*)
15	22	6.858	**School_distance**(*<1000;2000)*) >+< **Vehicle_type**(*Nakl_auto_naves*)
16	24	5.701	**School_distance**(*<2000;3000)*) >+< **Vehicle_type**(*Nakl_auto_naves*)
17	26	24.225	**School_distance**(*<3000;4000)*) >+< **Vehicle_type**(*Nakl_auto_naves*)
18	29	25.948	**School_distance**(*<4000;5000)*) >+< **Vehicle_type**(*Nakl_auto_naves*)
19	33	4.041	**School_distance**(*<5000;10000)*) >+< **Vehicle_type**(*Nakl_auto_naves*)
20	39	29.522	**School_distance**(*<10000;15000)*) >+< **Vehicle_type**(*Nakl_auto_naves*)
21	4	8.529	**School_distance**(*<200;300)*) >+< **Vehicle_type**(*Autobus*)
22	6	8.752	**School_distance**(*<300;400)*) >+< **Vehicle_type**(*Autobus*)
23	9	4.254	**School_distance**(*<500;600)*) >+< **Vehicle_type**(*Autobus*)
24	16	6.363	**School_distance**(*<800;900)*) >+< **Vehicle_type**(*Autobus*)
25	34	35.371	**School_distance**(*<5000;10000)*) >+< **Vehicle_type**(*Traktor*)
26	40	7.100	**School_distance**(*<10000;15000)*) >+< **Vehicle_type**(*Traktor*)
27	2	4.390	**School_distance**(*<100;200)*) >+< **Vehicle_type**(*Jizdni_kolo*)
28	10	5.147	**School_distance**(*<500;600)*) >+< **Vehicle_type**(*Jizdni_kolo*)
29	12	6.427	**School_distance**(*<600;700)*) >+< **Vehicle_type**(*Jizdni_kolo*)
30	14	9.431	**School_distance**(*<700;800)*) >+< **Vehicle_type**(*Jizdni_kolo*)
31	35	7.563	**School_distance**(*<5000;10000)*) >+< **Vehicle_type**(*Jizdni_kolo*)
32	1	69.313	**School_distance**(*<100*) >+< **Vehicle_type**(*Nezjisteno*)
33	3	180.077	**School_distance**(*<100;200)*) >+< **Vehicle_type**(*Nezjisteno*)
34	5	144.092	**School_distance**(*<200;300)*) >+< **Vehicle_type**(*Nezjisteno*)
35	7	138.452	**School_distance**(*<300;400)*) >+< **Vehicle_type**(*Nezjisteno*)
36	8	99.649	**School_distance**(*<400;500)*) >+< **Vehicle_type**(*Nezjisteno*)
37	11	79.035	**School_distance**(*<500;600)*) >+< **Vehicle_type**(*Nezjisteno*)
38	13	18.550	**School_distance**(*<600;700)*) >+< **Vehicle_type**(*Nezjisteno*)
39	15	36.019	**School_distance**(*<700;800)*) >+< **Vehicle_type**(*Nezjisteno*)
40	17	42.523	**School_distance**(*<800;900)*) >+< **Vehicle_type**(*Nezjisteno*)
41	19	5.650	**School_distance**(*<900;1000)*) >+< **Vehicle_type**(*Nezjisteno*)

Obrázek 5.15: Výsledné hypotézy pro vzdálenost od školy (CHI, subset).

5.8 Využití geografických dat II – nemocnice

Říká se, že blízkost nemocnice pomáhá zachránit život díky rychlejšímu zásahu. To znamená, že u nehod, které se stávají blíže nemocnic, je menší pravděpodobnost smrti některého z jejich účastníků. Analytická otázka v tom případě zní:

> *Existuje u silničních nehod závislost mezi vzdáleností od nemocnice a tím, zda při nehodě nebo v jejím bezprostředním následku někdo zemřel?*

V tom případě „v bezprostředním následku“ znamená v době do 24 hodin. Očekávaným výsledkem je závislost taková, že čím blíže nemocnice se nehody dějí, tím pravděpodobnější je, že nejsou to nehody smrtelné.

Použitá nastavení

Úloha byla spuštěna s těmito parametry:

- procedura: 4ft-Miner,
- antecedent: délka 1, atribut *Hospital_distance_1km* s maximálním počtem kategorií rovným 1,
- sukcedent: délka 1, atribut *Dead_number* s vybranou pevnou kategorií „0“, v první etapě s negativním charakterem, ve druhé a třetí etapě s pozitivním a negativním charakterem,
- podmínka: prázdná, ve třetí etapě délka 1, atribut *Dead_and_Major* s vybranou pevnou kategorií „0“ a negativním charakterem,
- kvantifikátory: v první a druhé etapě BASE 10, ve třetí etapě BASE 0, v první etapě AAD 0,1 a BAD 0,1, ve druhé a třetí etapě CHI $\alpha = 0{,}01$ a CHI $\alpha = 0{,}1$.

Negativní charakter atributu znamená, že se jeho hodnota bere s logickou negací. V případě této úlohy jsou tedy v příslušných etapách brány v potaz nehody, ve kterých počet mrtvých (nebo vážně zraněných) není nulový.

Výsledky úloh

První etapou bylo spuštění úlohy s kvantifikátory AAD a BAD. Kvantifikátor AAD měl za úkol nalézt intervaly vzdálenosti, pro které jsou smrtelné nehody častější, naopak BAD ty, ve kterých jsou tyto nehody méně časté. Na obr. 5.16 se nachází výsledné seznamy hypotéz, tentokrát seřazeny podle hodnoty parametru *AvgDf*.

V seznamu výsledků použití kvantifikátoru AAD je vidět, že hypotézy netvoří žádnou smysluplnou posloupnost. Nelze hovořit o vztahu mezi vzdáleností a relativním počtem smrtelných nehod. Na druhou stranu je také vidět, že ve většině intervalů, pro které je vzdálenost od nemocnic větší než 2 000 m, je relativní počet smrtelných nehod vyšší než na celém souboru dat. Seznam na obr. 5.16b potvrzuje, že se v případě vzdáleností menších než 2 000 m

jedná naopak o silně podprůměrné zastoupení smrtelných nehod. Dvě ostatní hypotézy doplňují obraz celkové neuspořádanosti výsledků a znemožňují nalezení jasného vztahu.

Z toho důvodu jsem přistoupil k druhé etapě a podobně jako v předchozí úloze (viz 5.7) jsem použil kvantifikátor CHI, a to jednou pro počet mrtvých větší než 0 ($\alpha = 0{,}1$), jednou pro rovný 0 ($\alpha = 0{,}1$). Výsledky se nachází na obr. 5.17. Je vidět, že se v seznamu hypotéz pro počet mrtvých rovný 0 nachází pouze vzdálenost od nemocnice menší než 2 000 m, přičemž pokud se jedná o vzdálenost do 1 000 m, tak je hodnota parametru *ChiSq* výrazně větší. Naopak ve výsledcích pro počet mrtvých větší než 0 je v seznamu opět vidět neuspořádanou posloupnost hypotéz pokrývajících téměř všechny ostatní kategorie. Lze na tomto základě konstatovat, že neexistuje žádný jednoduše vyjádřitelný vztah mezi vzdáleností od nemocnice a počtem smrtelných nehod, avšak ve vzdálenosti do 2 000 m jsou tyto nehody podstatně vzácnější a je to ovlivněno právě touto vzdáleností.

Poslední etapou bylo doplnění úlohy o podmínku, pomocí nichž se braly v potaz pouze vážné nehody, tzn. ty, ve kterých alespoň jeden člověk byl těžce raněn nebo mrtev. Díky tomu se posoudil vliv vzdálenosti od nemocnice na šanci přežití vážné nehody. Výsledné hypotézy se nachází na obr. 5.18. Hodnota parametru *ChiSq* je v případě nehod bez obětí vyšší než ve druhé etapě, naopak v případě smrtelných nehod je obecně menší než před přidáním podmínky, intervaly opět nejsou uspořádány. Lze tedy jednoznačně říct, že blízkost nemocnice velmi výrazně ovlivňuje šanci na přežití účastníků nehody.

Na těchto pokusech je vidět, že vzdálenost od nemocnice je parametrem, na kterém lze provádět dataminingové úlohy a dosahovat výsledků. Ve spojení s výsledky předchozí úlohy to vede k závěru, že je použití GIS a geografických údajů v dolování znalostí z databází užitečné. Na základě toho mohu tedy doporučit budoucí provádění výzkumu tímto směrem.

Nr.	Id	AvgDf	Hypothesis
1	11	0.743	**Hospital_distance_1km**(*<17000;18000)*) >+< **~Dead_number**(*0*)
2	4	0.571	**Hospital_distance_1km**(*<7000;8000)*) >+< **~Dead_number**(*0*)
3	7	0.442	**Hospital_distance_1km**(*<10000;11000)*) >+< **~Dead_number**(*0*)
4	6	0.385	**Hospital_distance_1km**(*<9000;10000)*) >+< **~Dead_number**(*0*)
5	10	0.327	**Hospital_distance_1km**(*<15000;16000)*) >+< **~Dead_number**(*0*)
6	8	0.319	**Hospital_distance_1km**(*<12000;13000)*) >+< **~Dead_number**(*0*)
7	5	0.206	**Hospital_distance_1km**(*<8000;9000)*) >+< **~Dead_number**(*0*)
8	2	0.169	**Hospital_distance_1km**(*<3000;4000)*) >+< **~Dead_number**(*0*)
9	9	0.165	**Hospital_distance_1km**(*<14000;15000)*) >+< **~Dead_number**(*0*)
10	1	0.147	**Hospital_distance_1km**(*<2000;3000)*) >+< **~Dead_number**(*0*)
11	3	0.115	**Hospital_distance_1km**(*<6000;7000)*) >+< **~Dead_number**(*0*)

a) Hypotézy získané použitím kvantifikátoru AAD.

Nr.	Id	AvgDf	Hypothesis
1	3	-0.223	**Hospital_distance_1km**(*<11000;12000)*) >+< **~Dead_number**(*0*)
2	4	-0.236	**Hospital_distance_1km**(*<13000;14000)*) >+< **~Dead_number**(*0*)
3	2	-0.420	**Hospital_distance_1km**(*<1000;2000)*) >+< **~Dead_number**(*0*)
4	1	-0.651	**Hospital_distance_1km**(*<1000*) >+< **~Dead_number**(*0*)

b) Hypotézy získané použitím kvantifikátoru BAD.

Obrázek 5.16: Výsledné hypotézy pro vzdálenost od nemocnice.

Nr.	Id	Chi-Sq	Hypothesis
1	1	47.324	**Hospital_distance_1km**(*<1000*) >=< **Dead_number**(*0*)
2	2	16.420	**Hospital_distance_1km**(*<1000;2000)*) >=< **Dead_number**(*0*)

a) Hypotézy pro nehody bez obětí.

Nr.	Id	Chi-Sq	Hypothesis
1	4	15.475	**Hospital_distance_1km**(*<7000;8000)*) >=< **-Dead_number**(*0*)
2	7	6.241	**Hospital_distance_1km**(*<10000;11000)*) >=< **-Dead_number**(*0*)
3	11	5.170	**Hospital_distance_1km**(*<17000;18000)*) >=< **-Dead_number**(*0*)
4	6	5.047	**Hospital_distance_1km**(*<9000;10000)*) >=< **-Dead_number**(*0*)
5	8	3.232	**Hospital_distance_1km**(*<12000;13000)*) >=< **-Dead_number**(*0*)
6	10	2.504	**Hospital_distance_1km**(*<15000;16000)*) >=< **-Dead_number**(*0*)
7	2	2.132	**Hospital_distance_1km**(*<3000;4000)*) >=< **-Dead_number**(*0*)
8	1	1.670	**Hospital_distance_1km**(*<2000;3000)*) >=< **-Dead_number**(*0*)
9	5	1.599	**Hospital_distance_1km**(*<8000;9000)*) >=< **-Dead_number**(*0*)
10	9	0.880	**Hospital_distance_1km**(*<14000;15000)*) >=< **-Dead_number**(*0*)
11	3	0.696	**Hospital_distance_1km**(*<6000;7000)*) >=< **-Dead_number**(*0*)

b) Hypotézy pro smrtelné nehody.

Obrázek 5.17: Výsledné hypotézy pro vzdálenost od nemocnice (CHI, bez podmínky).

Nr.	Id	Chi-Sq	Hypothesis
1	1	17.497	**Hospital_distance_1km**(*<1000*) >=< **Dead_number**(*0*) / **-Dead_and_Major**(*0*)
2	2	3.858	**Hospital_distance_1km**(*<1000;2000)*) >=< **Dead_number**(*0*) / **-Dead_and_Major**(*0*)

a) Hypotézy pro nehody bez obětí.

Nr.	Id	Chi-Sq	Hypothesis
1	1	2.958	**Hospital_distance_1km**(*<3000;4000)*) >=< **-Dead_number**(*0*) / **-Dead_and_Major**(*0*)
2	4	2.687	**Hospital_distance_1km**(*<12000;13000)*) >=< **-Dead_number**(*0*) / **-Dead_and_Major**(*0*)
3	6	2.454	**Hospital_distance_1km**(*<17000;18000)*) >=< **-Dead_number**(*0*) / **-Dead_and_Major**(*0*)
4	2	2.342	**Hospital_distance_1km**(*<7000;8000)*) >=< **-Dead_number**(*0*) / **-Dead_and_Major**(*0*)
5	5	1.939	**Hospital_distance_1km**(*<14000;15000)*) >=< **-Dead_number**(*0*) / **-Dead_and_Major**(*0*)
6	8	1.568	**Hospital_distance_1km**(*<20000;21000)*) >=< **-Dead_number**(*0*) / **-Dead_and_Major**(*0*)
7	3	0.753	**Hospital_distance_1km**(*<10000;11000)*) >=< **-Dead_number**(*0*) / **-Dead_and_Major**(*0*)
8	7	0.742	**Hospital_distance_1km**(*<19000;20000)*) >=< **-Dead_number**(*0*) / **-Dead_and_Major**(*0*)

b) Hypotézy pro smrtelné nehody.

Obrázek 5.18: Výsledné hypotézy pro vzdálenost od nemocnice (CHI, s podmínkou).

5.9 Shrnutí

Při práci s databází nehod jsem realizoval několik desítek různých úloh. V této kapitole jsem uvedl osm jejich vybraných typů, popsal způsob jejich realizace a prezentoval výsledky. Úlohy jsem rozdělil do třech hlavních skupin: hledání asociačních pravidel, frekvenční analýzy a využití geografických dat. V každé skupině jsem prezentoval více různých variant, abych čtenáři ukázal široké spektrum možností při využití systému LISp-Miner. U každého typu úlohy jsem také představil jeho charakteristické rysy důležité při volbě nastavení úlohy.

Asociační pravidla jsem rozdělil do třech typů: pravidla bez podmínky, pravidla s podmínkou a složitější pravidla. Jako výsledek prvního typu jsem vybral hypotézy týkající se vztahu mezi číslem silnice a typem vozidla. Tímto způsobem jsem určil mj. silnici III/1027, na které se nehody motocyklů dějí nadprůměrně často. Výsledky hypotéz s podmínkou tvoří mj. skutečnost, že nehody služebních automobilů, při kterých byl řidič pod vlivem alkoholu, se konají nejčastěji v ranních hodinách, naopak soukromé automobily za stejných podmínek havarují nejméně často. V rámci této úlohy jsem taky ukázal, že některé hypotézy nemají smysl, přestože splňují podmínky úlohy. V posledním typu úlohy jsem uvedl hypotézy hovořící o blízkém vztahu příčiny nehody a specifického místa jejího konání, např. nedodržení bezpečné vzdálenosti v blízkosti přechodu pro chodce.

Frekvenční analýzu jsem opět rozdělil do třech druhů úloh: hledání monotónních posloupností, hledání malého rozptylu frekvencí a hledání velkého rozptylu frekvencí. V rámci úlohy prvního typu jsem prezentoval nalezené parametry, pro které počet nehod v letech klesal, nebo stoupal. Příkladem klesající posloupnosti jsou počty nehod, při kterých byl řidič pod vlivem alkoholu, rostoucí pak nehody, při kterých byl řidič nemocný či po úrazu. Výsledkem druhého typu úloh bylo nalezení nehod, jejichž frekvence se během týdne příliš nemění (oproti celkovému histogramu, na kterém je vidět výrazný pokles o víkendu). Jsou to např. nehody konající se v oblouku či srážky se sloupem. Třetí typ úlohy generoval hypotézy, jejichž počet v jednotlivých měsících výrazně kolísal (oproti celkovému histogramu, na kterém rozdíly nejsou výrazné). Příkladovým výsledkem jsou nehody, při kterých byl řidič unaven.

Poslední skupinu tvořily dva typy úloh: založené na vzdálenosti od školy a založené na vzdálenosti od nemocnice. V rámci prvního typu jsem našel vztah mezi typem vozidla a vzdáleností (blízko škol častěji není zjištěn typ vozidla). Výsledkem úlohy druhého typu bylo potvrzení, že v blízkosti nemocnice je šance na přežití těžké nehody mnohem vyšší.

Tyto výsledky mi dovolují učinit dva hlavní závěry. Prvním je potvrzení toho, co tvrdí mnoho dřívějších výzkumných prací (zmínil jsem některé z nich v úvodu), totiž že je použití metod data miningu na datech z oblasti nehodovosti přínosné a dává velmi zajímavé výsledky. Díky těmto postupům lze hledat jak obecná pravidla, tak i partikulární fenomény. Přínosem je zejména možnost automatického zpracování objemného souboru dat, jakým 800 000 záznamů o nehodách bezpochyby je. Výsledky takového zpracování jsou navíc při použití vhodného softwaru přehledné a agregované do čitelných skupin. Na základě toho mohu silně doporučit pokračování v tomto výzkumu.

Druhý závěr se týká využití geografických informačních systémů v dobývání znalostí z nehodových dat. Použití GIS pro odvození nových údajů a jejich následné zpracování pomocí dataminingových úloh umožnilo odhalení nových vztahů nebo potvrzení všeobecně známých znalostí. Na tomto základě mohu konstatovat, že další výzkum s využitím této metody je slibný a je záhodno v něm pokračovat.

Závěr

Dolování znalostí z databází je dnes velmi rozšířeným způsobem zpracování dat. Rovněž data v oblasti nehodovosti jsou kvůli potenciálně velmi přínosným výsledkům zpracovávána v mnoha zemích světa. První studie tohoto typu byly realizovány už před 20 lety a metody výzkumu se od té doby neustále vyvíjejí, čehož důkazem jsou studie zmíněné v úvodu. Hlavním cílem této práce bylo navázání na světový trend a aplikace metod data miningu na data o nehodách, které se odehrály v České republice. V druhé řadě bylo cílem posouzení možnosti využití geografických informačních systémů v dataminingovém zpracování. K naplnění těchto cílů jsem použil databázi nehod, které se v letech 2007–2013 staly na území Středočeského kraje.

Oproti původnímu zadání se první část práce věnuje podrobnému popisu metody GUHA a systému LISp-Miner namísto klasické rešerše používaných dataminingových algoritmů a programů. Důvodem pro tuto změnu je především větší přínosnost pro čtenáře. Díky podrobnému popisu zvoleného systému a metody, na které je tento systém založen, může čtenář jednoduše pokračovat ve výzkumu, který jsem v rámci této práce realizoval. To by nebylo možné, pokud by namísto podrobného shrnutí několika způsobů zpracování byl uveden podstatně stručnější přehled existujících možností. Takovýto přehled navíc existuje ve formě samostatného článku [3], nebyl by tedy žádným přínosem. Samotná volba systému vyplývá z jeho volné dostupnosti, z neustálého vývoje a zejména z formy vhodné k první zkušenosti s data miningem (sledování metodiky CRISP-DM).

Druhá část práce se věnuje všemu, co badatel musí vykonat před zahájením samotného hledání zákonitostí v datech, tedy přípravě dat a teoretickým radám ohledně práce se softwarem určeným k realizaci úloh data miningu. Na příkladu zde použitých dat je podrobně popsána etapa předzpracování dat. Popsána jsou samotná data a veškeré úkony provedené za účelem převedení dat do formy vhodné pro zpracování. Tyto úkony se dělí do třech skupin: čištění i úpravy dat, rozšiřování dat o odvozené údaje a definování atributů v systému LISp-Miner. Každá skupina je popsána tak, aby se čtenář podrobně seznámil se způsobem realizace dané etapy. V případě tvorby atributů se proto jedná spíše o popis existujících přístupů a funkcí než o popis provedené práce – samotná tvorba atributu je totiž krátkým a jednoduchým úkolem, ovšem pouze za předpokladu, že badatel ví, čeho chce dosáhnout.

Dále jsou v této části uvedeny nejdůležitější možnosti systému LISp-Miner a pravidla, která je vhodné dodržovat během práce. Práce s dataminingovým softwarem může být zpočátku velmi obtížná, zejména pro nezkušeného uživatele. K definování analytické otázky, na jejímž základě bude vytvořena úloha, je mnohdy nutno použít nástroje frekvenční analýzy atributů. Samotná příprava úlohy často vyžaduje využití mnoha specifických nastavení pro přesnou realizaci analytické otázky. Ani interpretace výsledků není v případě dolování zna-

lostí z dat zcela jednoznačná ani jednoduchá, jelikož velmi snadno lze v důsledku drobné chyby učinit nepravdivé závěry. Z těchto důvodů jsem v této kapitole shrnul všechny poznatky a zkušenosti, které jsem získal během své práce se systémem LISp-Miner. Umožní to budoucím uživatelům vyvarovat se slepých uliček a vyřešit pravděpodobné problémy.

Třetí část popisuje provedený výzkum a prezentuje jeho výsledky. Tato část je stěžejní částí celé práce. Prezentovány jsou v ní tři hlavní typy úloh: úlohy hledání asociačních pravidel, úlohy frekvenční analýzy a úlohy využití geografických dat. Popis jednoho typu tvoří popis několika různých podtypů. V případě asociačních pravidel se jedná o pravidla bez podmínky, s volitelnou podmínkou a komplexní pravidla, v případě frekvenční analýzy jsou to úlohy hledání monotónních posloupností, malých rozptylů hodnot a velkých rozptylů hodnot. Geografická data byla využita v rámci hledání asociačních pravidel a obsahují údaje o vzdálenosti od škol i o vzdálenosti od nemocnic.

V rámci popisu každé úlohy je v první řadě nastolena analytická otázka, která byla základem pro definici dané úlohy. Dále jsou podrobně popsány použité parametry, přičemž nově použitá nastavení jsou vždy opatřena vysvětlujícím komentářem. Následně jsou popsány a interpretovány výsledky úlohy. V závislosti na charakteru dané úlohy jsem uvedl buď všechny, nebo pouze vybrané hypotézy, přičemž ve druhém případě jsem se vždy řídil požadavkem, aby výběr ukazoval co nejširší spektrum možných výsledků. Všechny prezentované hypotézy jsem doplnil komentářem a provedl jsem jejich interpretaci. Díky tomu může tato kapitola být pro čtenáře inspirací pro jeho vlastní výzkum, což jsem si v úvodu stanovil jako jeden z cílů této práce práce.

Z širšího hlediska lze na základě výsledků dolování znalostí z databáze nehod formulovat dva závěry. Prvním je potvrzení přínosnosti aplikace metod data miningu na podrobná nehodová data. Dosažené výsledky mj. poukazují na obecné zákonitosti, které se v datech vyskytují, např. na výraznou tendenci ke zvyšování počtu nehod způsobených únavou řidiče v letním období. Dále byla také vyjádřena řada vztahů mezi jednotlivými charakteristikami nehod, např. velmi vysoká frekvence nehod motocyklů na některých silnicích či větší podíl nehod vozidel hromadné dopravy v pondělí. Lze proto říci, že dolování znalostí z dat v oblasti silniční nehodovosti je velmi slibným směrem výzkumu a zcela jistě stojí za to v tomto výzkumu pokračovat.

Druhým závěrem je kladné vyhodnocení možnosti využití GIS v úlohách DZD. Tento přístup umožnil přiřazení všem nehodám nejbližšího bodu daného typu, vypočtení vzdálenosti od tohoto bodu a následně hledání závislostí mezi touto vzdáleností a jinými charakteristikami nehod pomocí metod data miningu. Ve výsledku byl vyjádřen vztah mezi vzdáleností od školy a frekvencí nehod, při kterých nebyl zjištěn typ vozidla, a také bylo v reálných datech potvrzeno, že blízkost nemocnice silně snižuje pravděpodobnost, že těžká nehoda bude smrtelná. Je to pouze nejjednodušší způsob využití potenciálu analytických funkcí GIS, avšak už na základě toho lze konstatovat, že další výzkum využívající spojení těchto dvou nástrojů může přinášet dosud nedosažené poznatky. V případě využití pokročilejších nástrojů (např. síťové a prostorové analýzy) nebo použití GIS pro interaktivní zobrazení na mapě nehod, které splňují podmínky dané nalezenými hypotézami, mohou být výsledky ještě zajímavější.

Úlohy prezentované v této práci jsou dle mých znalostí prvním pokusem o zkoumání zákonitostí v databázi nehod v českém prostředí. Lze je v nejlepším případě považovat za základ, na kterém bude v budoucnu stavěn širší a hlubší výzkum. Ne všechny z 65 na databázi definovaných atributů byly použity v rámci zpracování. Prvním směrem, jakým lze na tuto

práci navázat, je proto zkoumání zákonitostí založených na ostatních parametrech, např. na hodnotě finanční škody. V databázi je k dispozici alespoň 50 nezávislých parametrů, což dává obrovský počet kombinací, které je ještě možné prozkoumat. Lze také uvažovat o vyžádání alespoň některých údajů o účastnících nehody (např. věku) od PČR, jelikož tyto údaje byly z databáze odstraněny před zpřístupněním jejího obsahu. Toto by mohlo vést k velmi zajímavým objevům. Dalším krokem by měla být také práce s daty z ostatních krajů České republiky a také s úplným souborem dat (tedy z celé ČR). Některé z úloh prezentovaných v této práci, zvláště např. hledání vztahu mezi číslem silnice a typem vozidla (popsané v sekci 5.1), zcela jistě přinesou zajímavé výsledky i při takovémto použití.

Druhým směrem, ve kterém stojí za to pokračovat, je rozšíření spektra geografických dat využívaných k hledání hypotéz. Na jednu stranu mohou to být jiné geografické lokace (např. hasičská zbrojnice), na druhou mohou to být i zcela jiné typy geografických dat. V této práci byly použity jenom data typu *point*, avšak za těmito účely lze použít i data typu *line* či *polygon*. Je možné např. zkoumat, zda se nehody v lese (tedy nehody, jejichž geografická poloha je uvnitř polygonu reprezentujícího les) stávají v určité době nebo z určitého důvodu (např. jsou to srážky se zvěří), lze také zkoumat např. vztah mezi frekvencí nehod způsobených nedostatečnou viditelností kvůli mlze a vzdáleností od nejbližší řeky. Je zde velký prostor, který stojí za to prozkoumat.

Velmi přínosné může být také zkombinování dat o nehodách s jinými daty, např. s údaji o počasí. Lze očekávat, že se povede najít např. výrazný vliv teploty na frekvenci nehod způsobených únavou řidiče, možná dokonce ve spojení s rokem výroby vozidla (starší automobily s větší pravděpodobností nebudou vybaveny klimatizací). Ještě odvážnější může být hledání závislostí mezi výraznými poklesy na finančních burzách a nehodami drahých automobilů (které pravděpodobně patří bohatým investorům), mezi počtem nehod způsobených alkoholem a konáním lokálních oslav atd. Zde jsou perspektivy velmi široké a zároveň mohou být vztahy, které by se podařilo najít, velmi neočekávané.

Ještě jinou možností je využití způsobu zpracování dat s využitím „okénka“, navrženého v mé bakalářské práci [10], k hledání oblastí, ve kterých se dané parametry vyskytují častěji. Definování kategorií na údajích o poloze a jejich následné využití v rámci úlohy může vést k identifikaci oblastí, které si zaslouží větší pozornost, např. k nalezení konkrétních míst, ve kterých jsou nehody způsobeny únavou řidiče velmi časté. Za tímto účelem lze v systému LISp-Miner také využít modulu „Geodata analysis“, který vznikl na základě zmíněné bakalářské práce.

Nejzásadnějším pokračováním této práce je však reálná aplikace poznatků, které už byly a mohou být jako výsledek dalšího výzkumu získány. I v rámci těch několika úloh, které jsou zde prezentovány, se nachází několik zajímavých výsledků, které mohou sloužit jako podnět k učinění kroků vedoucích ke zvýšení bezpečnosti na některých silnicích (např. silnice II/102). Bylo by možné snadno vytvořit seznam všech českých silnic, na kterých se nadprůměrně často stávají nehody daného typu vozidla, např. jízdních kol. Tento seznam by pak bylo možné předat institucím spravujícím tyto silnice (lze ho např. jednoduše rozdělit na silnice první, druhé a třetí třídy), což by mohlo ve výsledku vést k reálnému zvýšení bezpečnosti dopravy v České republice. Tyto potenciální přínosy by měly být tou největší motivací pro badatele, kteří mají zájem o navázání na tuto práci a chtějí pokračovat v dobývání znalostí z databází nehod.

Literatura

[1] FLACH, Peter, BLOCKEEL, Hendrik, GÄRTNER, Thomas aj. On the Road to Knowledge. Mining 21 years of UK traffic accident reports. In: MLADENIĆ, Dunja aj. *Data Mining and Decision Support: Integration and Collaboration*. London: Kluwer Academic Publishers Boston, Mass., 2003, s. 143-156. ISBN 1-4020-7388-7.

[2] SARAEE, Mohamad, KERRY, Jonathan, LLOYD, Michelle, MARKEY, Christine. Data mining application: case of road traffic accidents in the UK West Midlands area 2000. In: *IC-AI*. Las Vegas: CSREA Press, 2004, s. 1102–1108. ISBN: 1-932415-32-7.

[3] JAYASUDHA, K., CHANDRASEKAR, C. An Overview of Data Mining in Road Traffic and Accident Analysis. In: *Journal of Computer Applications*, 2009, **2**(4), 32–37. ISSN 0974-1925.

[4] KRISHNAVENI, S., HEMALATHA, M. A Perspective Analysis of Traffic Accident using Data Mining Techniques. In: *International Journal of Computer Applications*, 2011, **23**(7), 40–48. ISSN 0975–8887.

[5] ANASTASI, Giuseppe, ANTONELLI, Michela, BECHINI, Alessio aj. Urban and social sensing for sustainable mobility in smart cities. In: *Sustainable Internet and ICT for Sustainability, SustainIT 2013, Palermo, Italy, 30-31 October, 2013, Sponsored by the IFIP TC6 WG 6.3 "Performance of Communication Systems"*, IEEE, 2013, 1–4. ISBN 978-3-901882-56-2.

[6] BESHAH, Tibebe, EJIGU, Dejene, ABRAHAM, Ajith, KRÖMER, Pavel, SNÁŠEL, Václav. Knowledge Discovery from Road Traffic Accident Data in Ethiopia: Data Quality, Ensembling and Trend Analysis for Improving Road Safety. In: *Neural Network World*, 2012, **22**(3), 215–244. ISSN 1210-0552.

[7] ŠIMŮNEK, Milan. *LISp-Miner. Šestnáct let vývoje akademického systému pro dobývání znalostí z databází*. Habilitační práce v oboru „Informatika“. Praha: VŠE-FIS, 2011.

[8] RAUCH, Jan, ŠIMŮNEK, Milan. *Dobývání znalostí z databází, LISp-Miner a GUHA*. Praha: Vysoká škola ekonomická, Nakladatelství Oeconomica, 2015. ISBN 978-80-245-2033-9.

[9] CHAPMAN, Pete a kol. *CRISP-DM 1.0. Step-by-step data mining guide* [online]. SPSS Inc., 2000 [cit. 22.05.2015]. Dostupné z: <`ftp://ftp.software.ibm.com/software/analytics/spss/support/Modeler/Documentation/14/UserManual/CRISP-DM.pdf`>

[10] URBANIEC, Krzysztof Paweł. *Dobývání znalostí z dopravních databází.* Bakalářská práce ve studijním programu „Technika a technologie v dopravě a spojích“, obor „Automatizace a informatika“. Praha: ČVUT FD, 2013.

[11] HÁJEK, Petr, HAVEL, Ivan, CHYTIL, Metoděj. GUHA - metoda systematického vyhledávání hypotéz. *Kybernetika*, 1966, **2**(1), 31–47.

[12] BURDA, Michal. *Získávání znalostí z databází - Asociační pravidla* [online]. Ostrava: VŠB-TU, 2004 [cit. 22.05.2015]. Dostupné z: <`http://www.fit.vutbr.cz/study/courses/ZZD/public/seminar0304/GUHA-text.pdf`>

[13] Domovská webová stránka projektu LISp-Miner (v anglickém jazyce) [online]. Katedra informačního a znalostního inženýrství, Fakulta informatiky a statistiky, Vysoká škola ekonomická v Praze, 2015 [cit. 22.05.2015]. Dostupné z: <`http://lispminer.vse.cz/`>

[14] BERKA, Petr, RAUCH, Jan, TOMEČKOVÁ, M. Data Mining in Aetherosclerosis Risk Factor Data. In: BERKA, Petr, RAUCH, Jan, ZIGHED, Djamel Abdelkader. *Data Mining and Medical Knowledge Management: Cases and Applications.* London: Information Science Reference, 2009, s. 376-397. ISBN 978-1-60566-218-3.

[15] RAUCH, Jan, RAŚ, Zbigniew W. aj. SEWEBAR Tinnitus – Project Presentation. In: *Znalosti 2009: zborník príspevkov z konferencie.* Bratislava: Vydavateľstvo Slovenskej technickej univerzity, 2009, s. 359-363.

[16] Domovská webová stránka projektu OpenStreeMaps [online]. Open Knowledge Foundation, 2015 [cit. 22.05.2015]. Dostupné z: <`http://www.openstreetmaps.org/`>

[17] Podstránka domovské webové stránky projektu Geofabrik věnovaná datům z České republiky [online]. Geofabrik GmbH, 2015 [cit. 22.05.2015]. Dostupné z: <`http://download.geofabrik.de/europe/czech-republic.html`>

[18] Domovská webová stránka programu PostgreSQL [online]. The PostgreSQL Global Development Group, 2015 [cit. 22.05.2015]. Dostupné z: <`http://www.postgresql.org/`>

[19] Domovská webová stránka programu Quantum GIS [online]. Faunalia, 2015 [cit. 22.05.2015]. Dostupné z: <`http://www.qgis.org/`>

[20] Domovská webová stránka modulu PostGIS [online]. PostGIS Project Steering Committee, 2015 [cit. 22.05.2015]. Dostupné z: <`http://postgis.net/`>

[21] Domovská webová stránka projektu mapy.cz [online]. Seznam.cz, a.s., 2015. [cit. 22.05.2015]. Dostupné z: <`http://www.mapy.cz/`>

Příloha A

Návody k přípravě softwarového prostředí

V této příloze se nachází návody, podle kterých doporučuji připravit softwarové prostředí určené ke zpracování dat způsobem popsaným v této práci. Seznam a krátký popis použitých programů se nachází v sekci 3.2.

A.1 Načtení databáze nehod do PostgreSQL

Prvním úkolem je vytvoření nové databáze na serveru localhost v programu PostgreSQL. Za tímto účelem se k serveru přihlásíme (měl by být už nastaven – pokud ne, vytvoříme ho podle návodu k programu) pomocí dvojího kliknutí na levém panelu. Následně pravým tlačítkem klikneme na seznam databází a vybereme ze seznámu „Nová databáze“. Ve zobrazeném okně databázi pojmenujeme a zadáme vlastníka (standardně „postgres“). Od tohoto okamžiku můžeme pracovat a zadávat SQL dotazy.

Pozor! V případě práce s více databázemi je potřeba mít v levém panelu v PostgreSQL vždy označenou správnou databázi nebo jednu z jejich součástí, aby příkazy nebyly vykonávány jinde, než bychom chtěli.

Pro načtení dat je potřeba vytvořit tabulku, která bude obsahovat příslušné sloupce. To se provede pomocí dotazu

```
CREATE TABLE nazev_tabulky (nazev_sloupce1 datovy_typ1,
nazev_sloupce2 datovy_typ2, ...);
```

zadaného v okně dotazů SQL. Následně je potřeba do tabulky zkopírovat údaje. K tomu jsou vhodné soubory v textovém formátu, ve kterých jsou data rozdělena např. čárkami či tabulátory. To se provede pomocí dotazu (nebo několika dotazů, pokud je souborů s daty více)

```
COPY nazev_tabulky (nazev_sloupce1, nazev_sloupce2, ...)
FROM 'adresa_souboru.' (FORMAT pouzity_format, HEADER, ENCODING typ_kodovani,
DELIMITER 'znak_pouzity_pro_oddelovani');
```

Po načtení všech souborů je databáze naplněna daty a připravena k dalším etapám předzpracování.

A.2 Propojení QGISu s PostgreSQL

Propojení QGISu a PostgreSQL se provádí pomocí volně použitelného zásuvného modulu PostGIS (viz 3.2). Pro instalaci je potřeba dodržovat návody uvedené na stránkách projektu. Po úspěšné instalaci PostGISu je potřeba PostgreSQL doplnit o nově instalovaný modul. To se provede po zadání dotazu

```
CREATE EXTENSION postgis;
```

do okna dotazů SQL. Po vykonání dotazu je možno používat nové rozšíření.

Pro přístup k databázi vytvořené v PostgreSQL je v QGISu potřeba navázat spojení s touto databází. Za tímto účelem je potřeba na levé liště v okénku s lokacemi (měly by tam být zobrazeny disky, např. C:, a ostatní položky, mj. PostGIS) kliknout pravým tlačítkem myši na položku PostGIS a vybrat možnost „Nové připojení“ (popř. v angličtině „Connect“). Ve zobrazeném okně je potřeba doplnit příslušné hodnoty. Položku „Hostitel“ není potřeba vyplňovat, pokud je PostgreSQL postaven na stejném počítači. Zbytek hodnot lze zjistit v PostgreSQL, jelikož je potřeba je vždy nastavit při tvorbě nové databáze a je potřeba je potom dodržet i v QGISu. Určitě je potřeba vyplnit název databáze, jméno uživatele a port, na kterém má probíhat komunikace. Nedoporučuji ukládat heslo, jelikož QGIS je ukládá v otevřeném textovém souboru. Pro ověření správnosti spojení je vhodné použít tlačítko „Test“.

Je potřeba si uvědomit, že i v případě, že test spojení proběhl správně, nemusí se nutně v QGISu nic zobrazit. Z toho důvodu nepřítomnost objektů ke zobrazení není znakem špatného propojení QGISu a PostgreSQL. QGIS nabídne objekty ke zobrazení pouze v případě, že jsou alespoň v jedné tabulce zvolené databáze uložena geografická data. Vytvoření správných sloupců je popsáno v sekci 3.3.4.

A.3 Propojení LISp-Mineru s PostgreSQL

Pro přístup LISp-Mineru k databázi vytvořené v PostgreSQL se využívá rozhraní ODBC. Drivery ODBC pro PostgreSQL je většinou potřeba nainstalovat. Informace o nich včetně odkazů na stránky se soubory ke stažení se nachází na oficiálních stránkách PostgreSQL na adrese http://odbc.postgresql.org/. Pro spuštění je potřeba vybrat z nabídky vhodnou pro pužívaný systém verzi driveru (v mém případě byla to verze 9.3.4-x64),stáhnout a dodržovat návod na instalaci.

Po instalaci je potřeba vytvořit nové DSN (Data Source Name). V systému Windows je k tomu určen Správce zdrojů dat ODBC. Ten se najde pomocí vyhledání fráze „ODBC“ v Ovládacích panelech. V případě Windowsu 7 se nachází ve složce „Nástroje pro správu“. Po jeho otevření je potřeba v záložce „Uživatelské DSN“ (popř. „Systémové DSN“) vybrat jeden ze dvou driverů: PostgreSQL ANSI nebo PostgreSQL Unicode. V českých podmínkách doporučuji použít driver Unicode. Po zvolení je potřeba konfigurovat nové spojení. V položce DataSource se stanoví zvolený název, v Database název databáze vytvořené v PostgreSQL, v případě databáze fungující na tomtéž počítači se do položky Server napíše „localhost“, zbytek tvoří hodnoty zvolené při dřívější tvorbě databáze. SSL Mode je zpravidla nastaven na hodnotu „disable“. Po kliknutí „Save“ je nové DSN vytvořeno.

Pozor! V případě používání 64bitového systému Windows je postup poněkud odlišný. Jelikož je LISp-Miner 32bitový program, je potřeba nutně spustit 32bitového Správce zdrojů dat ODBC. Ten se nachází v adresáři C:\Windows\System32\odbcad32.exe (v případě instalace systému Windows na jiném disku, použijte tento disk místo disku C:\ v uvedeném adresáři). Dále se postupuje podle dřívějšího popisu.

Po úspěšném vytvoření DSN lze přejít k vlastnímu propojení LISp-Mineru s PostgreSQL. V adresáři ...\LISp-Miner\DBCon\ (nebo v jiné složce, do které je nainstalovaný LISp-Miner) se nachází soubor LMEmpty.mdb – vzorová, prázdná metabáze. Tento soubor zkopírujeme do jiného adresáře, změníme jeho název na námi zvolený (např. Nehody.mdb) a takto upravený soubor vrátíme do původního adresáře (nebo do jakéhokoliv námi zvoleného adresáře, ve kterém např. chceme mít všechny soubory spojené s analyzovanými daty). Následně spustíme LISp-Miner pomocí souboru LMWorkspace.exe. Klikneme tlačítko „Associate an existing metabase with data“. Ve zobrazeném okně zaškrtneme možnost „Analyzed data available as:“, zvolíme „Existing ODBC data source“ a do kolonky „Data source“ zadáme název dříve vytvořeného DSN. Níže („LISp-Miner Metabase available as:“) zvolíme „MS Access file“, klikneme tlačítko „Browse“ a vybereme soubor .mdb, který jsme před chvíli vytvořili a uložili ve vybraném adresáři. Po kliknutí tlačítka „OK“ bude LISp-Miner připraven k práci.

Jinou možností je kliknout ve zobrazeném po spuštění LISp-Mineru okně tlačítko „New from DSN“. Následně je potřeba vyplnit příslušně název zdroje dat ODBC (okno pro zobrazování možností bude nejspíše prázdné, proto je ho potřeba vyplnit ručně), název souboru, ve kterém bude uložena metabáze, a adresář, do kterého bude uložena. Úplně dole je také potřeba vybrat, zda se jedná o uživatelské, nebo systémové DSN. Po kliknutí tlačítka „OK“ by měl být systém také připraven k práci.

Příloha B

Tvorba parametrů odvozených od geografických dat

Parametry odvozené od geografických dat představují hlavní přínos geografického informačního systému pro tuto práci. Jejich tvorba vyžaduje současného využití PostgreSQL a QGISu a znalosti v oblasti specifických PostGISovských funkcí.

V této sekci je popsán způsob odvození geografických dat. Hlavní důraz je kladen na vlastní technické provedení. Kromě toho se zde nachází odkazy na jednotlivé sloupce zmíněné v předchozí sekci, ve které jsou také vyjmenována všechna geografická data, která byla v rámci tvorby této práce použita.

B.1 Výběr typu geografických objektů

Prvním krokem při rozšiřování databáze o geografická data je vždy volba geografických objektů, které nás v daném okamžiku zajímají. Data OSM se skládají z několika sloupců, mj. „name“, „place“ či „other_tags“. Nejčastěji nás bude zajímat poslední zmíněný sloupec, ve kterém je uložen typ objektu (*type*). Seznam používaných typů (*tagů*) lze najít na stránkách projektu na adrese <`http://wiki.openstreetmap.org/wiki/Map_Features/`>.

Jelikož soubor pro jediný kraj obsahuje kolem milionu záznamů typu *point*, jeho prohlížení v QGISu není praktické. QGIS má tendenci se zasekávat při pokusu o zobrazení všech hodnot posledního sloupce, kvůli čemuž většinou není možné ručně prohledávat soubor. Pokud by měl čtenář zájem o prohlížení souboru .dbf v podobě tabulky, doporučuji za tímto účelem použít program Altap Salamander (<`http://www.altap.cz/`>).

Po zvolení typu objektu je potřeba zobrazit výběr žádaných objektů. To se provede pomocí SQL dotazu, který určí, jaké objekty mají být ve výběru zobrazeny. K tomu slouží okno „Vlastnosti vrstvy“ (pravé tlačítko myši na názvu vrstvy v levém spodním okně a možnost „Vlastnosti“) a jeho záložka „Obecné“. Ve spodní části okna se nachází tlačítko „Tvorba dotazů“. Po jeho kliknutí se zobrazí příslušné okno. Dotaz bude mít podobu:

```
"type" = 'nazev_objektu' (OR "type" = 'nazev_objektu2' OR ...)
```

Pokud chceme např. zobrazit jenom školy, dotaz bude vypadat takto:

```
"type" = 'school'
```

Název *type* může být samozřejmě nahrazen názvem jiného sloupce, nicméně všechny relevantní informace jsou v našem případě umístěny právě v tomto sloupci.

Po zadání dotazu je vhodné ho otestovat pomocí tlačítka „Test“. Pokud počet výsledných záznamů není nulový, můžeme tento dotaz aplikovat pomocí tlačítka „OK“. Pokud žádné hodnoty nebudou vraceny, nejspíš znamená to chybu v syntaxi a je potřeba dotaz přeformulovat.

Upozornění: Pokud chceme dotaz po otestování upravit, je vždy potřeba použít tlačítko „Vyčistit“ před zadáním nového dotazu. V opačném případě neumožní QGIS jeho opětovné otestování. Nejlepším řešením je uložení aktuálního obsahu do schránky, vyčištění okna tlačítkem „Vyčistit“, vložení obsahu ze schránky a následná úprava.

Po opětovném kliknutí „OK“ v okně vlastností budou ve zvolené vrstvě zobrazeny pouze objekty splňující dotaz. Takto upravenou vrstvu je potřeba uložit jako novou vrstvu. Za tímto účelem klikneme na vrstvu pravým tlačítkem myši a zvolíme možnost „Uložit jako“. Po zvolení názvu (je vhodné vyhnout se znaku „-“ či mezeře) je vrstva uložena a načtena do QGISu.

B.2 Import geografických dat do PostgreSQL

Zpracování dat v podobě úprav geometrie či tvorby odvozených atributů je mnohem lépe provádět v PostgreSQL než v QGISu, protože PostgreSQL pracuje s údaji v databázi podstatně rychleji a efektivněji než QGIS. Za tímto účelem je potřeba vrstvu vytvořenou podle postupu z předchozího odstavce importovat do příslušné databáze v PostgreSQL jako novou tabulku. Až pak je možné provádět potřebné úkony.

Před další prací musíme mít zapnutý PostgreSQL a inicializovanou příslušnou databázi. Pokud ji máme, na horní liště QGISu zvolíme záložku „Databáze“, ve které vybereme „Spit (Shapefile import)“ a klikneme „Import Shapefile do PostGIS“. Ve zobrazeném okně nejdříve připojíme databázi (pokud propojení popsané v A.2 bylo provedeno, měla by být v nabídce). Poté zvolíme název sloupce geometrie (opět je vhodné vyhnout se znaku „-“ či mezeře) a typ geometrie – druhý z nich musí mít pro data OSM hodnotu „4326“, což odpovídá geometrii WGS 84. Pod spodním oknem klikneme tlačítko „Přidat“ a vybereme vrstvu (či vrstvy), kterou chceme importovat. Po kliknutí tlačítka „OK“ bude import proveden.

Po samotném importu je potřeba ještě provést drobné formální úpravy v rámci PostgreSQL. Importované tabulky mají názvy s uvozovkami (podobně jako při přiřazování geometrických údajů nehodám, viz 3.3.4), které je potřeba pro pohodlnější práci odstranit. Totéž platí pro sloupec s geometrickými údaji. Provede se to pomocí dvojice dotazů:

```
ALTER TABLE "nazev_tabulky" RENAME TO nazev_tabulky;
ALTER TABLE nazev_tabulky RENAME COLUMN "nazev_sloupce" TO nazev_sloupce;
```

Po těchto úpravách je ještě potřeba převést sloupec s geometrií do formátu odpovídajícímu geometrii zpracovávaných dat. To se provede pomocí dvojice dotazů:

```
ALTER TABLE nazev_tabulky ADD nazev_sloupce geometry(datovy_typ,
                                                    kod_nove_geometrie);
UPDATE nazev_tabulky SET nazev_sloupce =
  ST_TRANSFORM(puvodni_sloupec_geometrie, kod_nove_geometrie);
```

V případě databáze nehod se jedná o geometrii S-JTSK (Křovákovo zobrazení), pro kterou je kód 102067. Datový typ může mít hodnotu „Point“, „lines“ nebo „multipolygons“. Příkladová konkrétní dvojice dotazů pro databázi nehod má tedy podobu:

```
ALTER TABLE StC_hospital ADD geom_Krovak geometry(Point,102067);
update StC_hospital set geom_krovak = ST_transform(geom_WGS,102067);
```

Po vytvoření nového sloupce lze původní sloupec odstranit, není to však k ničemu potřeba. Teď lze pokračovat v práci.

B.3 Odvození vzdálenosti od nejbližšího bodu

Prvním krokem musí být přirozeně vyznačení pro každý bod (v našem případě pro každou nehodu) nejbližšího pro něj objektu daného typu. Toho se dosáhne tak, že se spočte vzdálenost od každého bodu tohoto typu, tyto vzdálenosti se seřadí do rostoucí posloupnosti a vybere se její první člen. Vzdálenost lze také omezit shora za účelem zmenšení délky posloupnosti (např. na 10 km) nebo v případě, že jsou větší vzdálenosti zcela irelevantní. K nehodě se pak do nového sloupce přiřadí primární klíč záznamu odpovídajícího nalezenému objektu v tabulce, která tyto objekty obsahuje. SQL dotaz je tentokrát složitější a jejich dvojice má podobu (ve třetím a čtvrtém řádku se nachází možné omezení vzdálenosti, pro použití je potřeba odstranit znaky komentáře):

```
ALTER TABLE tabulka_dat ADD odkaz_na_objekt bigint;
UPDATE tabulka_dat
  SET odkaz_na_objekt = (SELECT klic_z_tabulky_objektu
    FROM tabulka_objektu
    --WHERE
    --ST_DISTANCE(tabulka_objektu.geometrie_objektu,
                                          tabulka_dat.geometrie_dat)<=10000
    ORDER BY tabulka_objektu. geometrie_objektu <-> tabulka_dat.geometrie_dat
    LIMIT 1);
```

***Poznámka**: V předposledním řádku se nachází drobné zjednodušení. Konstrukce datového typu odpovídajícího Křovákově zobrazení umožňuje přímé porovnání dvou hodnot míst porovnávání skutečné vzdálenosti. Chyba může nastat, pokud se vzdálenost mezi bodem a dvěma objekty liší o hodnotu v řádu desítek centimetrů. Tehdy může být tímto zjednodušeným způsobem vybrán objekt, který je ve skutečnosti o několik desítek centimetrů dále než jiný. Pro potřeby zpracování, jaké zde provádíme, to však nemá význam, můžeme si proto dovolit urychlit výpočet tímto způsobem. V případě potřeby lze však nahradit symbol „<->“ funkcí „st_ distance“ podle vzoru ve třetím řádku od konce.*

Po přiřazení každému řádku nejbližšího objektu přichází na řadu výpočet samotné vzdálenosti. Tento úkol není komplikovaný, protože objekt je jednoznačně přiřazený. Postačí jedině spočíst vzdálenost pomocí funkce „st_distance“ aplikované na geometrická data nehody a příslušného objektu a uložit výsledek do nového sloupce. Dvojice SQL dotazů bude mít tvar:

```
ALTER TABLE tabulka_dat ADD sloupec_vzdalenosti float;
UPDATE tabulka_dat SET sloupec_vzdalenosti=
ST_DISTANCE((SELECT geometrie_objektu FROM tabulka_objektu WHERE
tabulka_objektu.klic_z_tabulky_objektu=tabulka_dat.odkaz_na_objekt),
                                                        geometrie_dat);
```

Výsledné hodnoty jsou vzdálenosi všech záznamů od nejbližšího objektu daného typu v metrech.

Příloha C

Atributy definované na databázi nehod

Tabulka C.1: Seznam atributů definovaných na databázi nehod.

Název atributu	Sloupec	Počet kategorií
Accident_cause	p12	64
Accident_cause_less_cat	p12	6
Accident_cause-Driving	p12	15
Accident_cause-Malfunction	p12	13
Accident_cause-Overtaking	p12	10
Accident_cause-Priority	p12	12
Accident_cause-Speed	p12	8
Accident_localization	p22	9
Accident_type	6	9
Accident_type-moving_vehicles	p7	4
Alcohol_influence	p11	2
Bars_distance	bars_dist	19
Bars_distance-10m	bars_dist	40
Car_direction_or_location	p52	8
Car_manufacturer	p45a	94
Consequences_type	p9	2
Crossed_road_type	p39	6
Day_of_Week	phdatum (extrakt)	7
Dead_and_Major	ku_p13ab	8
Dead_number	ku_p13a_int	5
Deaths_and_injuries	ku_p13sum	7
Drift	p49	2
Driver_category	p55a	10
Driver_state	p57	10
External_driver_influence	p58	6

Tabulka C.1 Seznam atributu definovaných na databázi nehod (pokračování)

Název atributu	Sloupec	Počet kategorií
Financial_loss_at_vehicle	p53int	24
Financial_losses	ku_p14_int	38
Fuel_pump_distance	fuel_dist	16
Hospital_distance	hospital_dist	18
Hospital_distance_1km	spital_dist	24
Hour	ku_hour	24
Layout	p28	7
Local_traffic_control	p24	5
Major_injuries	ku_p13b_int	7
Material_leak	p50b	3
Minor_injuries	ku_p13c_int	8
Month	phdatum (extrakt)	12
Police_distance	police_dist	19
Police_distance_1km	police_dist	30
Post_office_distance	post_office_dist	17
Responsibility	p10	8
Road_category	p36	8
Road_number	p37int	1410
Road_size	p21	6
Road_state	p17	12
Salvaging	p51	3
School_distance	school_dist	18
School_distance-10m	school_dist	31
Solid_object_type	p8	9
Solid_object_type_no_X	p8	10
Specific_location	p27	10
Specific_location_no_X	p27	11
Surface_state	p16	10
Surface_type	p15	7
Traffic_control_type	p23	4
Vehicle_after_accident	p50a	5
Vehicle_age	ku_vehicle_age	26
Vehicle_characteristics	p48a	18
Vehicle_type	p44	17
Vehicle_year_of_production	p47int	60
Vehicles_number	p34int	11
Visibility	p19	7
Vision	p20	7
Weather	p18	8
Year	phdatum (extrakt)	7

Příloha D

Použité SQL dotazy

Tato příloha obsahuje přesné znění SQL dotazů použitých ve fázi předzpracování dat. Popis jednotlivých úkonů se nachází v kapitole 3. Pořadí sekcí odpovídá této kapitole.

V některých dotazech je vidět název tabulky „nehody_sc“, v jiných „nehody_text“. Je to způsobeno tím, že tabulka „nehody_sc“ obsahující pouze data ze Středočeského kraje byla vytvořena z tabulky „nehody_text“, která obsahuje všechna data, až po vyjmutí čísel kraje a okresu. Všechny dotazy lze aplikovat i na tabulku „nehody_text“.

Načtení souborů

```
CREATE TABLE nehody_text
(phid SERIAL, y text, x text, p1 text, p36 text, p37 text, p38 text,
p40 text, p41 text, p2a text, den text, cas text, p6 text, p7 text,
p8 text, p9 text, p10 text, p11 text, p12 text, p13a text, p13b text,
p13c text, p14 text, p15 text, p16 text, p17 text, p18 text, p19 text,
p20 text, p21 text, p22 text, p23 text, p24 text, p27 text, p28 text,
p34 text, p35 text, p39 text, p44 text, p45a text, p47 text, p48a text,
p49 text, p50a text, p50b text, p51 text, p52 text, p53 text, p55a text,
p57 text, p58 text, a text, b text, f text, g text, h text, i text, j text,
k text, l text, n text, o text, p text, q text, r text, s text, t text);

COPY nehody_text
(p1, p36, p37, p2a, den, cas, p6, p7, p8, p9, p10, p11, p12, p13a, p13b,
p13c, p14, p15, p16, p17, p18, p19, p20, p21, p22, p23, p24, p27, p28,
p34, p35, p39, p44, p45a, p47, p48a, p49, p50a, p50b, p51, p52, p53, p55a,
p57, p58, a, b, f, g, x, y, h, i, j, k, l, n, o, p, q, r, s, t)
FROM adresa_souboru (FORMAT CSV, HEADER, ENCODING 'win1250', DELIMITER ',');
```

Sjednocení formátu identifikačního čísla

```
UPDATE nehody_text
SET php1 = p1
WHERE p1 ~ '^[0-9]+$';

Update nehody_text
SET php1 = '0'|| p1
WHERE LENGTH (P1)=11;
```

Sjednocení formátu data konání nehody

```
ALTER TABLE nehody_text ADD COLUMN phdatum date;

UPDATE nehody_text
SET phdatum = to_date (p2a, 'YYYY-MM-DD')
WHERE p2a ~ '^[0-9]+-[0-9]+-[0-9]+$' ;

UPDATE nehody_text
SET phdatum = to_date (p2a, 'DD.MM.YYYY')
WHERE p2a ~ '^[0-9]+\.[0-9]+\.[0-9]+$';

UPDATE nehody_text
SET phdatum = to_date (p2a, 'DD/MM/YYYY')
WHERE p2a ~ '^[0-9]+/[0-9]+/[0-9]+$';

UPDATE nehody_text
SET phdatum = to_date (p2a, 'MM/DD/YYYY')
WHERE p2a ~ '^[0-9]+/[0-9]+/[0-9]+$'
AND phid IN
(SELECT phid FROM
(SELECT phid,p2a, to_date (p2a, 'DD/MM/YYYY') as datum,
(EXTRACT(dow FROM to_date(p2a, 'DD/MM/YYYY'))+1)::int % 7
as den_v_tydnu,den FROM nehody_text WHERE p2a ~ '^[0-9]+/[0-9]+/[0-9]+$')
as tbl
WHERE den_v_tydnu::int - den::int <> 1
AND den_v_tydnu::int - den::int <> 0
AND den_v_tydnu::int - den::int <> -6);

UPDATE nehody_text
SET phdatum = (date '1900-01-01'+p2a::integer* '1 day'::interval
-'2 day'::interval)::date WHERE p2a~ '^[0-9]+$';
```

Vyjmutí čísel kraje a okresu

```
alter table nehody_text add ku_hour integer;
alter table nehody_text add ku_hour integer;

update nehody_text
set ku_okres=cast(substr(php1, 3,2) as integer);

update nehody_text
set ku_kraj=cast(LEFT(php1, 2) as integer);
```

Přidání sloupců s geometrickými daty

```
ALTER TABLE nehody_text ADD COLUMN phx double precision;
ALTER TABLE nehody_text ADD COLUMN phy double precision;

UPDATE nehody_text
SET phx = replace( replace (x ,',',',.'),'"','' )::double precision
WHERE replace( replace (x ,',',',.'),'"','' )~ '^-{0,1}\d*\.{0,1}\d+$';

UPDATE nehody_text
SET phx = - phx
WHERE phx>0;

UPDATE nehody_text
SET phx = NULL
WHERE phx NOT BETWEEN -904539 AND -431680;

UPDATE nehody_text
SET phy = replace( replace (y ,',',',.'),'"','' )::double precision
WHERE replace( replace (y ,',',',.'),'"','' )~ '^-{0,1}\d*\.{0,1}\d+$';

UPDATE nehody_text
SET phy = - phy
WHERE phy>0;

UPDATE nehody_text
SET phy = - phy
WHERE phy NOT BETWEEN -1227290 AND -935232;

INSERT INTO spatial_ref_sys (srid, auth_name, auth_srid, proj4text, srtext)
VALUES ( 102067, 'esri', 102067,
'+proj=krovak +lat_0=49.5 +lon_0=24.83333333333333 +alpha=
30.28813975277778 +k=0.9999 +x_0=0 +y_0=0 +ellps=bessel +units=m +towgs84=
498.17,136.89,510.08,6.007,4.343,3.831,3.38 no_defs <>',
```

```
'PROJCS["S-JTSK_Krovak_East_North",GEOGCS["GCS_S_JTSK",
DATUM["Jednotne_Trigonometricke_Site_Katastralni",SPHEROID
["Bessel_1841",6377397.155,299.1528128]], PRIMEM["Greenwich",0],
UNIT["Degree",0.0174532925199433295]],PROJECTIONKrovak?,
PARAMETER["False_Easting",0],
PARAMETER["False_Northing",0],PARAMETER["Pseudo_Standard_Parallel_1",78.5],
PARAMETER["Scale_Factor",0.9999],PARAMETER["Azimuth",30.28813975277778],
PARAMETER["Longitude_Of_Center",24.83333333333333],
PARAMETER["Latitude_Of_Center",49.5],
PARAMETER["X_Scale",-1],PARAMETER["Y_Scale",1],
PARAMETER["XY_Plane_Rotation",90],
UNIT["Meter",1],AUTHORITY["EPSG","102067"]]');

SELECT AddGeometryColumn ('public','nehody_text',
'the_GeomKrovak',102067,'POINT',2);

UPDATE nehody_text SET "the_GeomKrovak"= GeomFromEWKT('SRID=102067;POINT
(' || nehody_text.phx || ' ' || nehody_text.phy || ')');

alter table "the_GeomKrovak" rename to the_GeomKrovak;
```

Úprava času konání nehody

```
alter table nehody_sc add ku_hour integer;
update nehody_sc
set ku_hour=cast(LEFT(ku_time, 2) as integer);
```

Sjednocení formátu a úprava roku výroby vozidla

```
alter table nehody_sc add p47int integer;
update nehody_sc set p47int=cast(p47 as integer);

update nehody_sc set p47int=null where p47='xx' or p47='XX' or p47='  ';
update nehody_sc set p47int=cast('200'||p47 as integer) where length(p47)=1;
update nehody_sc set p47int=cast('19'||p47 as integer)
where cast(p47 as integer)>20 and length(p47)=2;
update nehody_sc set p47int=cast('20'||p47 as integer)
where cast(p47 as integer)<20 and length(p47)=2;
```

Úprava dalších parametrů na datový typ *integer*

Všechny parametry byly převedeny stejným způsobem, proto uvádím zde pouze dva z nich. Pro převedení ostatních parametrů byl pouze změněn název sloupce. V případě sloupce *ku_p14_int* (druhý dotaz) je navíc hodnota vynásobena číslem 100.

```
alter table nehody_sc add ku_p13c_int integer;
update nehody_sc set ku_p13a_int=cast(p13a as integer);

alter table nehody_sc add ku_p14_int integer;
update nehody_sc set ku_p14_int=cast(p14 as integer)*100;
```

Suma počtů raněných a mrtvých

```
alter table nehody_sc add ku_p13sum integer;
Update nehody_sc set ku_p13sum = ku_p13a_int+ku_p13b_int+ku_p13c_int;

ď»żalter table nehody_sc add ku_p13ab bigint;
update nehody_sc set ku_p13ab=ku_p13a_int+ku_p13b_int;
```

Stáří vozidla

```
alter table nehody_sc add ku_vehicle_age integer;
update nehody_sc set ku_vehicle_age=cast(left(cast(phdatum as text), 4)
as integer)+1-p47int;
```

Vzdálenost od vybraných objektu typu *point*

Tyto dotazy se nachází v příloze B.