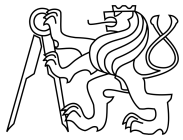
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ



ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Automatická rekonstrukce chybějících dat
Student:	David Záleský
Vedoucí:	doc. RNDr. Ing. Marcel Jiřina, Ph.D.
Studijní program:	Informatika
Studijní obor:	Teoretická informatika
Katedra:	Katedra teoretické informatiky
Platnost zadání:	Do konce zimního semestru 2017/18

Pokyny pro vypracování

1) Seznamte se s problémem měření nákupního chování osob v obchodě, kdy se sleduje jejich pohyb prostorem, dotýkání se zboží, a braní a vracení zboží z a do regálu.
2) Seznamte se s kamerovým systémem a měřicím procesem, který byl použit pro měření v prodejně Albert, kde byla získána reálná data a dále identifikujte možná úzká místa, která vedou ke ztrátě nebo poškození měřených dat.
3) Prozkoumejte formát dat a ukládané informace, které byly získány během měření. Proveďte analýzu těchto dat, zejména ze statistického hlediska.
4) Na základě porovnání skutečnosti, dostupných naměřených dat a provedených analýz navrhněte postup a dílčí metody, jak zrekonstruovat a doplnit chybějící údaje tak, aby co nejlépe odpovídaly skutečnosti.
5) Navržené metody implementujte po konzultaci s vedoucím práce ve vhodném jazyku.
6) Implementované metody ověřte na nezávislé sadě naměřených dat ze stejného měření, která jsou k dispozici, vyhodnoťte dosaženou přesnost a navrhněte možná zlepšení.

Seznam odborné literatury

Dodá vedoucí práce.

L.S.

doc. Ing. Jan Janoušek, Ph.D.
vedoucí katedry

prof. Ing. Pavel Tvrdík, CSc.
děkan

V Praze dne 7. března 2016

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra teoretické informatiky

Bakalářská práce

Automatická rekonstrukce chybějících dat

David Záleský

Vedoucí práce: doc. RNDr. Ing. Marcel Jiřina, Ph.D.

17. května 2016

Poděkování

Děkuji docentu Marcelu Jiřinovi za cenné rady, za ochotu při vedení práce a v první řadě vůbec za vypsání tohoto tématu.

Rovněž děkuji uživatelům serveru Stackoverflow.com, kteří se potýkali s podobnými problémy jako já, a uživatelům, kteří jejich problémy řešili.

Prohlášení

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

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené.

V Praze dne 17. května 2016 .....

České vysoké učení technické v Praze

Fakulta informačních technologií

Odkaz na tuto práci

Záleský, David. *Automatická rekonstrukce chybějících dat.* Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2016.

Abstrakt

Tato práce se zaměřuje na chyby vzniklé při měření chování zákazníků v prodejně obchodního řetězce. Popisuje průběh a metodiku měření, identifikuje druhy možných chyb, příčiny jejich vzniku a možná řešení. Řešení dále analyzuje z hlediska jejich použitelnosti vzhledem k poskytnutým datům. Vybraná řešení následně implementuje v programovacím jazyce Python.

Klíčová slova oprava chyb, rekonstrukce dat, obchodní řetězec, python

Abstract

This paper focuses on errors in measuring customers' activity in a supermarket. It describes the procedure of gathering data, identifies potential errors, their causes, and potential solutions. The solutions are then analyzed to determine their usability with provided data. Selected solutions are implemented in programming language Python.

Keywords fixing errors, data reconstruction, supermarket, python

Obsah

Seznam obrázků

Úvod

V silně konkurenčním prostředí obchodních řetězců jsou jejich provozovatelé vděčni za každou výhodu, kterou mohou získat. Jedním z často užívaných prostředků je i sledování chování zákazníků v prodejnách.

Podstatou tohoto sledování je zaznamenávání údajů o tom, kolik zákazníků prodejnu navštívilo, o jaké zboží se zajímali, a co nakonec koupili. Takto získané údaje jsou následně užity pro marketingové účely.

S rozvojem výpočetní technologie a jejím prorůstáním do všech oborů lidské činnosti dochází pochopitelně i k jejímu užití při sledování zákazníků. Vybrané části prodejen jsou tak osazovány snímacím zařízením, které požadované údaje automaticky zaznamenává, a takto získaná data jsou následně počítačově zpracovávána.

Při záznamu dat i při jejich zpracování však může docházet k chybám, které snižují věrohodnost dat samotných. A právě na tyto chyby je moje práce zaměřena.

V teoretické části práce přiblížím průběh celého procesu, a popíši chyby, které při něm vznikají. Chyby následně roztřídím do kategorií podle pravděpodobného místa vzniku a obtížnosti jejich nápravy.

V praktické části pak navrhnu metody, jakými lze chybám předcházet či alespoň omezit jejich dopad na správnost výsledků měření. Vybrané metody implementuji ve zvoleném programovacím jazyce a ověřím jejich účinnost.

Smyslem práce tedy je umožnit efektivnější analýzu naměřených dat tím, že omezím jejich chybovost, a tedy zvýším jejich vypovídající hodnotu.

Kapitola 1

Základní informace

1.1 Popis reálného použití

Podstatou projektu, který je základem pro tuto práci, je analýza spotřebitelského chování v prodejnách velkých obchodních řetězců prostřednictvím k tomu účelu určeného snímacího zařízení.

Měření probíhá tak, že vybrané regály jsou osazeny snímacími zařízeními (podrobněji o nich viz sekci 1.4), která snímají úzký prostor před sebou, a zaznamenávají jakýkoliv pohyb v oblasti.

Snímací zařízení jsou schopna monitorovat pohyb zákazníků ve vybraných částech prodejen. Kromě toho jsou však dostatečně přesná na to, aby z jimi pořízených záznamů bylo možné zjistit, jak přesně si zákazník počínal.

Při následné analýze je toto chování tříděno do pěti hlavních kategorií, které jsou později využívány při statistickém zpracování dat. Jedná se o tyto kategorie akcí:

- Pohyb v obchodě
- Zastavení se
- Sáhnutí do regálu
- Odebrání zboží z regálu
- Vrácení zboží do regálu

Každá z těchto akcí představuje informaci, která může být hodnotná při tvorbě marketingové politiky obchodního řetězce.

V této práci jsou používána data z reálného měření v období 16. 9. 2015 až 12. 10. 2015 v některých částech jedné z prodejen obchodního řetězce Albert.

1.2 Předchozí výzkum

Tento způsob monitorování chování zákazníků je ve světě poměrně unikátní, což naneštěstí znamená, že nejsou k dispozici žádné materiály, na které bych ve své práci mohl navazovat.

Výzkum tohoto typu prováděla například společnost GfK Czech[1], nicméně detaily procesu zpracovávání dat nejsou nikde zveřejněny. V této práci je tedy prováděna analýza používaných dat, převážně založená na datech samotných, spíše než na dokumentaci k nim.

1.3 Základní pojmy

Některá slova běžného jazyka mají v této práci speciální význam. Jejich běžný význam je sice všeobecně známý, nicméně v této práci jsou užívány většinou ve velmi specifickém významu. Je tedy nutné poskytnout jejich definice pro účely této práce.

1.3.1 Zákazník

Zákazníkem je zde myšlen člověk, který vstoupí do sekce, a to po celou dobu pobytu v ní. Při přechodu mezi moduly v rámci jedné sekce je stále považován za téhož zákazníka. V momentě, kdy zákazník opustí sekci, není možný jeho návrat. Pokud se tentýž člověk do sekce vrátí, je již považován za nového zákazníka bez znalosti předchozí historie.

1.3.2 Sekce

Sekce představuje uličku v obchodě. Ulička je tvořena regály se zbožím. V rámci jedné sekce se zpravidla jedná o zboží stejného druhu (např. šampóny, bonboniéry, etc.) Ulička je souvislá. Je tedy možné do ní vstoupit (či z ní naopak vystoupit) pouze na krajích.

Sekce se skládá z modulů.

1.3.3 Modul

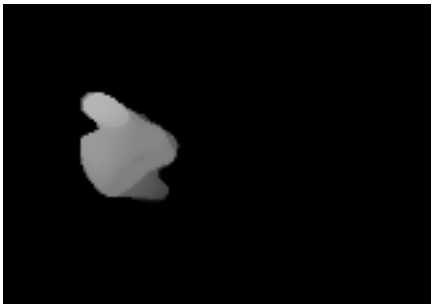
Modul je část sekce osazená jedním snímacím zařízením. Mezi jednotlivými moduly není žádné volné místo, které by zákazníkovi umožňovalo odchod ze sekce. Na každý modul v sekci tedy přímo navazuje jiný modul, pokud tento modul není na okraji sekce.

1.4 Snímací zařízení

Snímací zařízení je základním zdrojem dat. Je v každém modulu umístěno nad nejvyšším patrem regálu v rovině zboží.

Zařízení je vybaveno dvěma Kinect senzory[2]. Jeden z nich snímá prostor před sebou. Zaznamenává tak údaje o dění ve svém zorném poli (které přibližně pokrývá prostor modulu, v němž je umístěno, a části modulů sousedních.), a to ve formátu hloubkových map[3].

Obrázek 1.1: Hloubková mapa, na níž je vidět jeden zákazník.



Takto získané mapy jsou hlavním zdrojem informací o průchodu zákazníků sekcemi.

Druhý senzor zaznamenává všechna překročení roviny zboží (tedy hranice regálu.) Senzor snímá prostor přímo pod sebou a je schopný rozeznat objekt překračující hranici regálu v obou směrech (dovnitř i ven). Data z tohoto senzoru jsou klíčová při získávání informací o tom, jaké zboží si zákazníci prohlíželi či vzali z regálu.

Všechny záznamy ze snímacího zařízení jsou pochopitelně opatřeny přesnou časovou značkou (s přesností na milisekundy), což umožňuje zkoumání délky jednotlivých činností zákazníků i rozdíly v chování zákazníků v různých denních dobách.

1.5 Využitelnost získaných dat

Jak již bylo uvedeno, data získaná měřením chování zákazníků jsou následně řetězci využívána pro zefektivnění jejich marketingové strategie.

Informace, které získaná data poskytují, jsou velmi užitečné při hodnocení účinnosti rozložení zboží v prodejně s cílem maximalizace prodeje. V tomto ohledu je relevantní velké množství nejrůznějších ukazatelů, na nichž může obchodník svá rozhodnutí zakládat. Pro ilustraci uvádím některé z nich:

Nejnavštěvovanější sekce Informace, o tom, které sekce navštíví nejvíce zákazníků jsou důležité ze dvou důvodů: 1. Obchodník z nich velmi dobře zjistí, jaké zboží je populární. 2. V případě, že návštěvnost sekce přetrvává i po změně sortimentu, lze této znalosti využít při rozmísťování zboží.

Často prohlížené zboží U kterého zboží se zákazník často zastaví? Jak často si do vezme do ruky? A v kolika procentech případů nakonec skutečně skončí v košíku? Tyto informace jsou zásadní pro organizaci zboží v regálech, i pro rozhodnutím o tom, zda (a v jaké míře) některé zboží vůbec prodávat.

Změny v návštěvnosti během dne Jak se během dne mění návštěvnost jednotlivých sekcí a popularita zboží?

Počet vráceného zboží Jak často zákazníci vracejí zboží z košíku zpět do regálu? A vracejí ho tam, odkud ho vzali?

Rovněž je možné zkoumat, kolik zákazníků „odpadne“ v jednotlivých částech procesu (vstup do sekce, zastavení u regálu, prohlédnutí zboží, vložení zboží do košíku, zaplacení zboží), a snažit se s přihlédnutím k těmto údajům optimalizovat jednotlivé fáze nákupního procesu.

Kapitola 2

Zaznamenávané informace

Jediná data, která snímací zařízení zaznamenává přímo, jsou hloubkové mapy a čas. Hloubkové mapy však samy o sobě žádné relevantní informace neposkytují. K jejich získání je nutné data správným způsobem interpretovat.

Tato práce se však interpretací hloubkových map, a opravou chyb při ní vznikajících, nezabývá. Zdrojová data, s nimiž v této práci operuji, jsou již zpracovaná do formy popsané v sekci 2.2.

Pokud tedy ve zbytku práce budu mluvit o (zdrojových) datech či (zaznamenávaných) informacích, myslí se tím již zpracovaná data a interpretované informace. Pokud bude v některé části odkazováno na skutečná data (hloubkové mapy), bude to vždy explicitně uvedeno.

2.1 Typy informací

2.1.1 Čas

Čas je jediná přímo zaznamenávaná informace, která je i přímo použitelná (tedy nevyžaduje interpretaci.) S každou hloubkovou mapou je ukládána i časová značka dokumentující okamžik jejího vzniku. Časová značka je velmi přesná (na milisekundy), což zaručuje, že pro každou mapu bude unikátní.

Při převádění map na informace se časová značka pouze přejímá.

2.1.2 Pohyb

Informace o pohybu je základní informace. Právě na základě této informace je možné propojovat záznamy o zákaznících napříč moduly, což je zásadní pro určení skutečného počtu zákazníků, kteří sekci navštívili.

Podstatou této informace jsou údaje o tom, kdy zákazník vstoupil do modulu, odkud přišel (zda zleva či zprava) a jak daleko byl od snímacího zařízení v okamžiku příchodu. Tytéž údaje (čas, směr a vzdálenost od snímacího zařízení) jsou zaznamenávány i v okamžiku odchodu. Právě na základě časové a místní podobnosti mezi záznamy ze sousedních kamer jsou jednotlivé záznamy propojovány tak, aby z nich bylo možné vyčíst ucelené informace o průchodu zákazníka sekcí.

Za okamžik příchodu je považována časová známka první hloubkové mapy, na níž je zákazník zachycen. Okamžik odchodu vychází z poslední takové mapy.

Každý zákazník, který v rámci jedné sekce navštíví více než jeden modul, bude zachycen více záznamovými zařízeními. Jelikož tato zařízení mezi sebou nijak nekomunikují, je nutné tyto informace dodatečně spojit. Pokud tedy jeden modul zaznamená odchod zákazníka a sousední modul v tentýž čas zaznamená příchod zákazníka z předchozího modulu, budou tyto dva záznamy považovány za záznamy o témže zákazníkovi.

V případě, že by ve stejný čas zákazníků přecházelo mezi moduly více, je možné využít i jejich vzdálenost od kamery pro přesnější určení, který je který.

2.1.3 Zastavení

Pokud se zákazník v jednom modulu zastaví na déle než dvě sekundy, je tato skutečnost, včetně délky zastavení, zaznamenána.

Z hlediska analýzy chování zákazníků je dobré vědět, na kterých místech se zákazníci více zastavují. Může to svědčit o popularitě jednotlivého zboží (zvláště v kombinaci s informacemi o interakci se zbožím) nebo z těchto informací můžou být identifikovatelné některé typické charakteristiky sdílené větším počtem zákazníků.

Snímací zařízení tuto potřebu reflektují tím, že zastavování sledují. Limit dvou sekund nutný pro zahájení měření byl nastaven proto, aby eliminoval běžné odlišnosti v pohybech lidí, a aby tak byly skutečně zaznamenávány pouze situace, kdy se zákazník zastavil.

2.1.4 Manipulace se zbožím

Pokud zákazník nějak interaguje se zbožím v monitorovaném regálu, je i tato skutečnost zaznamenávána. A to ve třech formách: Sáhnutí do regálu, odebrání zboží z regálu a vrácení zboží do regálu.

Sáhnutí do regálu spočívá v tom, že zákazník, zpravidla rukou, překročí hranici regálu, a po nějaké době ji překročí v opačném směru. Klíčovým rozdílem oproti dalším dvěma formám manipulace se zbožím je zde to, že zákazník nic do regálu nevkládá, ani z něj neodebírá.

Odebrání zboží je charakteristické tím, že zákazník pronáší přes hranici regálu směrem ven něco, co nepronášel směrem opačným.

Vrácení zboží je pak opakem odebrání. Zákazník tedy pronáší přes hranici regálu směrem dovnitř něco, co následně nepronáší směrem ven.

Ve všech třech případech je u těchto typů informace je zaznamenáván čas (kdy k manipulaci došlo) a doba trvání (doba, po kterou byla překročena hranice regálu).

U akcí odebrání a vrácení zboží jsou navíc zaznamenávány i souřadnice místa kontaktu s hranicí regálu (místa akce).

Souřadnice jsou specifikovány dvojicí hodnot. První z nich udává vzdálenost místa akce od levé strany regálu v centimetrech. Druhá hodnota udává pořadí poličky v regálu. Dolní polička má číslo 1, každá vyšší polička má číslo o jedna vyšší.

Tyto informace jsou zásadní pro statistickou analýzu chování zákazníků, a právě v nich spočívá podstata celého projektu.

2.2 Formát dat

Data získaná ze snímacích zařízení ve formě hloubkových map jsou zpracovávána do tabulkového formátu, jenž je jako zdroj dat používán v této práci. Tento formát je založen na takzvaných „akcích.“ Každá akce představuje nějaký triviální úkon zákazníka a je reprezentována jedním řádkem tabulky.

Samotné zpracovávání hloubkových map není předmětem této práce, a proto zde nebudou uváděny podrobnosti tohoto procesu.

2.2.1 Zdrojové soubory

Tabulky jsou zpracovávány pro každý modul zvlášť za každý den. Celkem je tedy za každý sledovaný den k dispozici tolik souborů s datovými tabulkami, kolik bylo ten den aktivních snímacích zařízení, tedy modulů.

Názvy datových souborů se skládají ze šesti částí:

1. Slovo „actions“ uvozuje názvy všech datových souborů. Je odkazem na povahu datových tabulek, které jsou vlastně tabulkami akcí.
2. Zkratka názvu sekce, do níž je modul zařazen. Jedná se o krátké písmenné označení, zpravidla prvních pět či méně písmen názvu sekce. První písmeno je velké, ostatní malá.
3. Číselné ID modulu.
4. Datum ve formátu MMDD.
5. Pokud je soubor prázdný, doplní se jeho název ještě o slovo „empty.“
6. Všechny datové soubory jsou ve ukládány ve formátu Microsoft Office Excel[4], tedy s příponou .xlsx

ID modulu a datum jsou spojeny podtržítkem, stejně jako datum a slovo empty, v případě, že je součástí názvu.

Příklad: Název souboru obsahujícího datovou tabulku z modulu 4 v sekci bonboniér z 15. září by byl `actionsBon4_0915.xlsx`.

Zdrojové soubory jsou organizovány do složek. Každá složka sdružuje soubory z jednoho modulu za všechny dny v rámci monitorovacího období. Název složky je tvořen plným názvem kategorie a číslem modulu spojených podtržítkem.

Pokud tedy máme k dispozici například data z deseti modulů za třicet dní, budou tato data obsažena ve 300 souborech členěných do deseti složek po třiceti prvcích.

2.2.2 Akce

Každá akce je charakterizována akčním kódem, který říká, o jaký druh akce se jedná. Akčních kódů je sedm. Tyto kódy přímo korespondují s výše uvedenými typy informací:

1. Vstup do modulu zvenčí (vstup do sekce)
2. Vstup do modulu ze sousedního modulu
3. Zastavení se
4. Sáhnutí do regálu
5. Odebrání zboží z regálu

6. Vrácení zboží do regálu
7. Odchod ze zorného pole kamery

2.2.2.1 Detaily

Každý řádek tabulky obsahuje následujících jedenáct sloupců v uvedeném pořadí:

Customer ID Identifikační označení zákazníka. Vytváří se v okamžiku příchodu zákazníka a je složené ze zákazníkovi vzdálenosti od snímacího zařízení a časové značky okamžiku jeho příchodu, což zaručuje jeho unikátnost v rámci modulu.

Matching ID Označení vytvářené v okamžiku odchodu na stejném principu jako Customer ID. Používá se k propojování dat napříč moduly. (Pouze pro akci odchodu – 7)

Sortiment Číselné označení sekce společné pro všechny kamery v dané sekci.

Module Unikátní číselné označení modulu (snímacího zařízení). Sousední moduly mají čísla po sobě bezprostředně následující.

Shelf number Pořadové číslo poličky v regálu počítáno od zdola. (Pouze pro akce odebrání zboží a vrácení zboží do regálu – 5; 6)

Distance Vzdálenost od levé strany regálu v centimetrech. (Pouze pro akce odebrání zboží a vrácení zboží do regálu – 5; 6)

Action code Akční kód. Viz výše.

Date Datum zahájení akce ve formátu MS Excel float.

Start time Čas zahájení akce ve formátu MS Excel float.

Action length Doba trvání akce v sekundách. (Pouze pro akce zastavení a manipulace se zbožím – 3; 4; 5; 6)

Enter/Exit module Číslo modulu, z něhož zákazník přišel či do nějž odešel. Pokud vstoupit do sekce, či z ní odešel, tak 0. (Pouze pro akce příchodu a odchodu – 1; 2; 7)

Pokud některé pole není akcí používáno, zůstává prázdné.
Action length a **Distance** jsou desetinná čísla. Ostatní čísla jsou přirozená.

Obrázek 2.1: Ukázka části datové tabulky.

	A	B	C	D	E	F	G	H	I	J	K
1	Customer ID	Matching ID	Sortiment	Module	Shelf number	Distance [cm]	Action code	Date	Start time	Action length[s]	Enter/Exit module
472	103_0928_135200_258	94_0928_135201_810	4	8			7	28.09.	13:52:01.810		0
473	42_0928_135205_696	unmatched	4	8			1	28.09.	13:52:05.696		0
474	42_0928_135205_696		4	8			3	28.09.	13:52:07.196	7.311	
475	42_0928_135205_696	102_0928_135215_507	4	8			7	28.09.	13:52:15.507		9
476	103_0928_135214_290	unmatched	4	8			1	28.09.	13:52:14.290		0
477	103_0928_135214_290		4	8	3	79	4	28.09.	13:52:29.426	1.06	
478	103_0928_135214_290		4	8			3	28.09.	13:52:15.790	22.606	
479	103_0928_135214_290	68_0928_135239_396	4	8			7	28.09.	13:52:39.396		9
480	96_0928_135237_068	unmatched	4	8			1	28.09.	13:52:37.068		0
481	96_0928_135237_068		4	8	5	112	4	28.09.	13:52:44.707	1.4	
482	96_0928_135237_068		4	8	5	105	6	28.09.	13:52:47.606	1.8	
483	96_0928_135237_068		4	8	3	76	4	28.09.	13:52:49.646	2.984	
484	96_0928_135237_068		4	8	3	77	6	28.09.	13:52:53.616	1.001	
485	96_0928_135237_068		4	8	3	89	4	28.09.	13:52:59.727	1.8	
486	96_0928_135237_068		4	8			3	28.09.	13:52:38.568	29.268	
487	96_0928_135237_068	73_0928_135308_836	4	8			7	28.09.	13:53:08.836		0
488	103_0928_135329_116	unmatched	4	8			1	28.09.	13:53:29.116		0
489	103_0928_135329_116		4	8			3	28.09.	13:53:30.616	2.711	
490	103_0928_135329_116	42_0928_135334_327	4	8			7	28.09.	13:53:34.327		0
491	95_0928_135653_445	unmatched	4	8			1	28.09.	13:56:53.445		0
492	95_0928_135653_445		4	8			3	28.09.	13:56:54.945	7.909	
493	95_0928_135653_445	93_0928_135703_854	4	8			7	28.09.	13:57:03.854		9
494	95_0928_135653_445	unmatched	4	8			1	28.09.	13:56:53.445		0
495	95_0928_135653_445		4	8			3	28.09.	13:56:54.945	7.909	
496	95_0928_135653_445	93_0928_135703_854	4	8			7	28.09.	13:57:03.854		9
497	95_0928_135653_445		4	8			3	28.09.	13:56:54.945	7.909	
498	95_0928_135653_445	93_0928_135703_854	4	8			7	28.09.	13:57:03.854		9
499	95_0928_135653_445	unmatched	4	8			1	28.09.	13:56:53.445		0
500	95_0928_135653_445		4	8			3	28.09.	13:56:54.945	7.909	
501	95_0928_135653_445	93_0928_135703_854	4	8			7	28.09.	13:57:03.854		9
502	95_0928_135653_445		4	8			3	28.09.	13:56:54.945	7.909	
503	95_0928_135653_445	93_0928_135703_854	4	8			7	28.09.	13:57:03.854		9

Kapitola 3

Práce s daty

Než se pustíme do samotné analýzy chyb, je nutné popsat základní metodu, která zde již byla několikrát zmíněna, a která je klíčová pro pochopení obsahu dat. Jedná se o metodu slučování dat o zákaznících napříč moduly.

Tato metoda umožňuje propojit akce zákazníka napříč moduly v rámci sekce, a poskytnout tak o něm ucelenější informaci.

3.1 Slučování dat o zákaznících

Poznámka: Následující popis metody předpokládá bezvadná data.

Při slučování se použijí u každého zákazníka akce příchodu do každého modulu (1; 2) a odchodu z něj (7). Každý záznam o akci 1, 2 či 7 má časovou značku. Rovněž uvádí i identifikační číslo modulu, ze kterého zákazník přišel, či do kterého odešel. Příchod do sekce či odchod z ní jsou označovány jako interakce s modulem 0.

Všechny záznamy z jednoho modulu o témže zákazníkovi používají stejné ID, začínají akcí příchodu a končí akcí odchodu. Všechny tyto akce se stejným ID (včetně akcí 3; 4; 5 a 6) budou po zbytek práce označeny jako „modulový záznam.“

Při slučování se tedy vytvoří uspořádané sekvence modulových záznamů tak, že každá z nich bude popisovat průchod jednoho zákazníka sekcí. Sekvence bude začínat modulovým záznamem s akcí příchodu z modulu 0 a bude končit modulovým záznamem s akcí odchodu do modulu 0.

Mezi těmito modulovými záznamy může být libovolný počet modulových záznamů takových, že časová značka akce odchodu z každého modulu se sho-

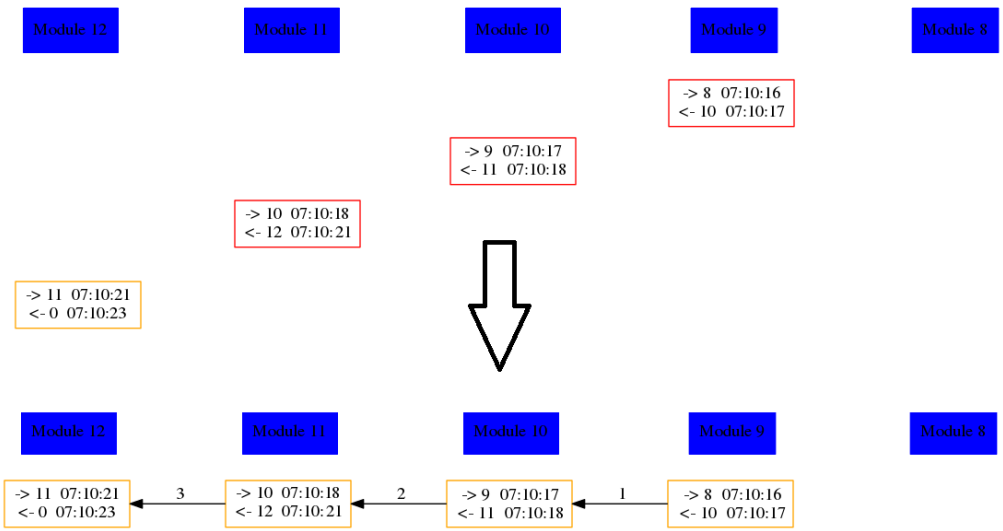
duje s časovou značkou akce příchodu do dalšího modulu v sekvenci. Rovněž pole Enter/Exit module u akce odchodu z každého modulu se shoduje s ID modulu následujícího modulového záznamu a pole Enter/Exit module u akce příchodu do u tohoto modulového záznamu se shoduje s ID modulu předchozího modulového záznamu.

Respektováním výše uvedených pravidel bude zajištěno, že všechny sekvence budou obsahovat unikátní zákazníky, a zároveň bude dosaženo minimálního počtu sekvencí.

Obrázek 3.1: Dva záznamy ze sousedních modulů (první dva řádky modul 4, druhé dva řádky modul 5), které popisují chování téhož zákazníka, který v 18:36:50 přešel z jednoho modulu do druhého.

Customer ID	Module	Action code	Date	Start time	Enter/Exit module
91_0923_121835_591	4	1	23.09.	18:34.59	0
91_0923_121835_591	4	7	23.09.	18:36:50	5
88_0923_183650_078	5	2	23.09.	18:36:50	4
88_0923_183650_078	5	7	23.09.	18:36:51	6

Obrázek 3.2: Grafické znázornění záznamů o jednom zákazníkovi ze čtyř sousedních modulů. Nahoře před sloučením, dole po něm. (Pro detailní popis tohoto typu grafu viz sekci 6.6.)



3.2 Pseudokód

Slučování zákazníků vypadá v pseudokódu následovně:

```
while records is not empty do
    for r in records do
        if r.origin_module == 0 then
            customers[r.ID] = new LinkedList();
            customers[r.ID].add(r);
            delete r from records;
            break;
        else
            for c in customers do
                if r.origin_module == c.destination_module &&
                r.origin_time == c.destination_time then
                    customers[c.ID].add(r);
                    delete r from records;
                    break;
                end
            end
        end
    end
end
```

Algoritmus 1: Metoda slučování dat o zákaznících

Metoda tedy funguje tak, že cyklicky prochází všechny modulové záznamy a porovnává je s aktuálními konci všech sekvencí, dokud nenajde shodu. Pokud shodu najde, zařadí modulový záznam do příslušné sekvence.

Pokud modulový záznam patří na začátek sekvence (akce příchodu z modulu 0), je zařazen do nové sekvence.

Vysvětlivky:

records je pole modulových záznamů ze všech modulů v sekci za jeden den

customers je pole sekvencí. Zpočátku prázdné, na konci běhu algoritmu obsahuje správně seřazené modulové záznamy.

origin_module je vlastnost modulových záznamů, která odpovídá na otázku, ze kterého modulu zákazník do aktuálního modulu přišel.

destination_module je vlastnost sekvencí modulových záznamů, která uvádí, do kterého modulu odešel zákazník v posledním záznamu v sekvenci.

Kapitola 4

Chyby v datech

V důsledku nedokonalostí v různých částech procesu získávání a zpracovávání dat, je nutné vzít v úvahu existenci potenciálních chyb. Právě zkoumání a řešení těchto chyb představuje základní náplň této práce.

Podle místa vzniku se chyby dělí na

1. Chyby vzniklé v důsledku nedostatků snímacího zařízení
2. Chyby při převádění dat ze snímacího zařízení do datové tabulky

V následující části bude rozepsáno, jaké konkrétní typy chyb byly při analýze snímacího procesu a dat samotných objeveny, a jaké jsou jejich příčiny.

4.1 Chyby snímacího zařízení

Tyto chyby vyplývají z povahy a kvality snímacího zařízení, a není možné se jim vyhnout bez výměny technologie. Za situace, kdy lepší technologie není z finančních důvodů k dispozici, je nutné se s nimi smířit a snažit se efektivně řešit jejich následky.

4.1.1 Nesynchronizovanost

Snímací zařízení nejsou během snímacího procesu nijak propojena, a tak každé měří čas individuálně. Rovněž kvalita vnitřních hodin jednotlivých zařízení není příliš valná, a hodiny na různých měřících zařízeních se různou mírou zpožďují či předbíhají. Ani počáteční synchronizace není bezvadná – na počátku každého měřícího dne byly časy na všech zařízeních „relativně“ synchronizované. To však v tomto případě znamená, že jsou synchronizovány s tolerancí +- 2 sekundy.

V Důsledku výše uvedeného není možné spoléhat se na přesnost časových značek. Například situace, kdy je příchod zákazníka do modulu zaznamenán dříve, než jeho odchod z předchozího modulu, je celkem běžná.

Není ani možné určit relativní odchylku času mezi jednotlivými moduly a čas přepočítat, jelikož míra zrychlení či zpoždění se v čase mění neznámým způsobem, a navíc byly hodiny během některých dnů opakovaně „synchronizovány“, aniž by o tom byly vedeny jakékoliv záznamy.

4.1.2 Slepá místa

Zorné pole snímacích zařízení nedosahuje až k pevné překážce na druhé straně uličky.

Je tedy možné, že zákazník projde kolem modulu, aniž by byl zachycen, nebo že opustí zorné pole (a je tedy považován za odešlého), přestože v modulu zůstává.

Tyto chyby mohou způsobovat zvláštní chování spočívající v tom, že o zákazníkovi bude záznam o odchodu z jednoho modulu a nebude o něm záznam o příchodu do sousedního modulu, nebo bude chybět záznam o průchodu jedním modulem.

4.1.3 Výpadky

Občas se stane, že snímací zařízení po nějakou dobu vůbec nefunguje v důsledku technické poruchy. Takovéto období se nazývá výpadkem. Po dobu výpadku nejsou zařízením zaznamenávána žádná data.

Důsledkem výpadků je pochopitelně chybějící informace o průchodu zákazníka „vypadlým“ modulem. Pokud bychom existenci výpadků neuvažovali, byla by takto poškozená data špatně interpretována. Místo toho, aby byla považována za data o jednom zákazníkovi procházejícím uličkou, by vypadala jako data o dvou zákaznících, z nichž jeden skončil svoji cestu ve vypadlém modulu, zatímco ten druhý svoji cestu v onom modulu započal.

4.2 Chyby při tvorbě datové tabulky

Tyto chyby jsou zpravidla důsledkem nedokonalostí při analýze hloubkových map získaných snímacím zařízením. V případě jejich řádné identifikace a popisu je tedy možné, že budou v některé z vylepšených verzí konverzního softwaru odstraněny, a nebude nutné je nadále uvažovat.

Některé z chyb však mohou být z části způsobeny i snímacím zařízením (bude se jednat hlavně o nedokonalosti hloubkových map), a pro ně bude platit totéž, co pro chyby z předchozí části.

4.2.1 Chybějící záznamy o zákaznících

Kromě globálních výpadků na hardwarové úrovni se v datech rovněž objevují lokální výpadky. Tedy situace, kdy v některém modulu nejsou zaznamenány informace o zákazníkovi, který do modulu na základě dat ze sousedních modulů prokazatelně ze sousedního modulu vstoupil, případně z něj i do (dalšího) sousedního modulu odešel.

Manuální analýzou hloubkových map bylo v naprosté většině zkoumaných případů zjištěno, že tyto chyby jsou důsledkem nedokonalostí konverzního softwaru, neboť na oněch mapách bylo vždy možné identifikovat zákazníka, jak ve správný čas prochází příslušným modulem.

Rovněž bylo zjištěno, že chyby tohoto druhu jsou relativně časté (více než čtvrtina zákazníků z map nebyla správně zaznamenána do tabulky). Naproti tomu opačné případy (zákazník je v tabulce, ale není v mapě) jsou naprosto ojedinělé (respektive spíše jen teoretické, neboť ve zkoumaném vzorku nebyl nalezen žádný.)

Konverzní software je tedy až příliš opatrný při analýze hloubkových map. A z obavy, aby neoznačil za zákazníka něco, co jím není, chybuje v mnoha případech opačným způsobem.

4.2.2 Duplicity

V datových tabulkách se velmi často objevují duplicitní řádky. (To jest řádky, které mají stejná všechna pole.) Tyto řádky nejsou nezbytně nutně po sobě jdoucí (spíše naopak.)

Záznamové zařízení neumožňuje záznam duplicitních dat, jelikož opatřuje každou akci velmi přesnou časovou značkou. Není tedy možné, aby jakékoliv dvě akce mohly dostat značku stejnou, a tedy být duplicitní. Jakékoliv duplicity tedy musejí vznikat v důsledku nedokonalosti konverzního programu.

Většina duplicitních akcí je rovněž logicky nemožná (například opakovaný příchod či odchod.) Jejich přijetí by tedy vedlo nejen k nesprávným datům, nýbrž i k datům nesmyslným.

Manuální analýza hloubkových map ukazuje, že duplicitní akce skutečně nastaly, nicméně pouze jednou, bez opakování. Nebyla zjištěna žádná pravi-

delnost v tom, které akce se v tabulce nesprávně vyskytují vícenásobně.

4.2.3 Vícenásobný odchod

Akce odchodu (akční kód 7) je charakteristická rovněž tím, že v rámci jedné návštěvy modulu ji zákazník může učinit pouze jednou. Po odchodu totiž modul již není schopný rozpoznat téhož zákazníka při jeho případném návratu, a považuje ho vždy za zákazníka nového. Následné slučování pak probíhá až na úrovni sekce.

V některých případech se však v rámci jednoho modulu u jednoho zákazníka akce odchodu vyskytuje vícekrát (zpravidla dvakrát). Tyto případy se od případů duplicitních řádků (vizte sekci 3.2.2) odlišují tím, že se u nich neshodují cílové moduly. Zákazník tedy z modulu zdánlivě odchází do obou sousedních modulů (případně současně odchází do sousedního modulu a opouští sekci.)

Obrázek 4.1: Ukázka části datové tabulky popisující jednoho zákazníka, u nějž se vyskytuje vícenásobný odchod.

Customer ID	Action code	Date	Start time	Action length[s]	Enter/Exit module
43_0930_104200_845	2	30.09.	10:42:01		8
43_0930_104200_845	3	30.09.	10:42:02	16.233	
43_0930_104200_845	7	30.09.	10:42:20		8
43_0930_104200_845	7	30.09.	10:42:20		10

4.2.4 Chyby v akcích manipulace se zbožím

Akce sáhnutí do regálu (akční kód 4), vzetí zboží z regálu (akční kód 5) a vrácení zboží do regálu (akční kód 6) jsou často zpracovávány chybně. Tyto chyby jsou pravděpodobně důsledkem nedokonalého konverzního algoritmu a vedou k velmi nepředvídatelnému chování.

V důsledku těchto chyb je možné, že kterákoliv akce manipulace se zbožím bude vyhodnocena jako kterákoliv jiná akce manipulace se zbožím. (Tedy akce 4 jako akce 5 či 6, akce 5 jako akce 4 či 6 a akce 6 jako akce 4 či 5.)

Některé z těchto chyb lze z dat identifikovat, neboť jsou porušením přirozeného běhu věcí (akce 6 před akcí 5 je velmi nepravděpodobná,) u ostatních však nic takového není možné, a to ani v kombinaci z daty ze sousedních modulů.

Praktický experiment se snímacím zařízením potvrdil vysokou četnost těchto chyb. Na druhou stranu však ukázal, že úplné ignorování akce, a její nezanesení do datové tabulky (byť případně i jako akci jinou), je vzácné.

4.3 Jiné chyby ve zdrojových souborech

V některých zdrojových souborech se ve zdrojové tabulce vyskytují neočekávané znaky, a to převážně mimo základních jedenáct sloupců. Tyto znaky mohou působit problémy při čtení dat ze souboru.

Soubory, které jsem měl při vypracovávání práce k dispozici, byly rovněž špatně pojmenovány. Datum v názvu souboru neodpovídalo datům akcí uvnitř souboru. Konkrétně bylo datum v názvu vždy o jeden den napřed oproti datům v souboru.

Kapitola 5

Oprava chyb

V předchozí části byly představeny některé chyby, jimiž mohou být zdrojová data stižena. V této části budou nastíněny možnosti jejich řešení, tak aby mohly sloužit jako základ pro implementaci.

Chyby zde budou rozděleny do tří kategorií odlišných od teoretického dělení. Bude se jednat o

1. Snadno odstranitelné chyby. Tedy chyby, které jsou z dat jednoznačně rozpoznatelné a je možné zcela eliminovat jejich dopad na zdrojová data.
2. Obtížněji odstranitelné chyby. To jest takové chyby, které ke svému odstranění vyžadují modifikaci metody slučování zákazníků.
3. Neodstranitelné chyby. Respektive chyby, jež není možné detekovat či odstranit pouze z dat obsažených v datových tabulkách.

5.1 Snadno odstranitelné chyby

Do této kategorie patří chyby, které je možné z dat snadno identifikovat, a je možné je odstranit již při čtení a prvotním zpracovávání dat.

5.1.1 Chyby ve struktuře datové tabulky

Při čtení dat je potřeba být co nejobezřetnější. Není možné předpokládat nic o znacích mimo prvních jedenáct sloupců tabulky, a jelikož tyto nejsou z hlediska dat relevantní, je vhodné je zcela ignorovat.

Přečtená data je rovněž nutno ošetřovat. Základní kontrola datových typů by však měla vyřešit většinu problémů, jelikož data jsou výstupem z jiného

programu, nikoliv uživatelským vstupem, a není tedy nutné je zabezpečovat nejvyšší možnou měrou.

Vstupní data, která neprojdou testem validity, bude ideální neuvažovat.

5.1.2 Odstraňování duplicit

Jelikož je jisté, že duplicitní data jsou důsledkem chyby, není k nim přihlíženo. Každý opakující se řádek bude uvažován pouze jednou, a na jeho kopie nebude brán zřetel.

Pro odstranění duplicit není nutné znát žádné datové závislosti, ba dokonce není nutné datům ani rozumět. Proto je ideální jejich odstranění provést jako jeden z prvních kroků. Při navrhování žádné další metody tak již nebude duplicity nutné uvažovat, což značně zjednoduší celý proces.

Samotný způsob odstraňování duplicit není nutné nijak podrobně popisovat, neboť je téměř samozřejmý. Řádky tabulky se jednoduše seřadí, a následně se sloučí všechny identické řádkové skupiny, stejně jako to dělá nativní součást unixového shellu, program uniq[5].

5.2 Obtížněji odstranitelné chyby

Do této skupina spadá většina popsaných chyb. Vyznačují se tím, že je zpravidla není možné z dostupných dat zcela detekovat či zcela odstranit.

Rovněž tyto chyby spojuje skutečnost, že jejich odstraňování probíhá formou modifikace metody slučování zákazníků.

5.2.1 Výpadky

Data z vypadlých modulů jsou rekonstruována za použití dat z okolních modulů. Pokud data ukazují, že nějaký zákazník do tohoto modulu vešel, a jiný zákazník z něj v přijatelném čase vyšel, jsou tito zákazníci sloučeni.

Data ze sousedních modulů však neumožňují rekonstruovat akce zastavení a manipulace se zbožím, neboť tyto se z nich nijak vyčíst nedají. Pro jejich rekonstrukci se zde používají statistické údaje ze stejného modulu v obdobnou denní dobu v jiné dny. Například, pokud modul vypadne ve středu od 9:00 do 10:00, použijí se zprůměrované informace z téhož modulu v pracovní dny v tomtéž čase. (Samozřejmě za dny, kdy modul neměl výpadek.)

Pro řešení výpadků se do metody slučování zákazníků přidává jeden krok oproti původní verzi. Pokud se stane, že žádný z modulových záznamů již není

možné zařadit do žádné sekvence, cyklus se ukončí, a začne nová fáze.

Tato fáze je podobná fázi předchozí, nicméně při určování, zda modulový záznam patří do sekvence se vypadlý modul přeskakuje, a porovnávání se provádí pouze mezi fungujícími moduly.

V pseudokódu tato nová fáze vypadá následovně:

```
while records is not empty do
    for r in records do
        if there is no active module on the same side of the isle as
        r.origin_module then
            customers[r.ID] = new LinkedList();
            customers[r.ID].add(r);
            delete r from records;
            break;
        else
            for c in customers do
                if there is no active module between r.origin_module and
                c.destination_module && r.origin_time
                =~c.destination_time then
                    customers[c.ID].add(r);
                    delete r from records;
                    break;
                end
            end
        end
    end
end
```

Algoritmus 2: Metoda slučování dat o zákaznících

Takto vytvořené sekvence však již na sebe nenavazují (chybí záznamy z přeskočených modulů). Před analýzou dat pro marketingové využití je tedy důležité chybějící data doplnit aproximací podle dat z obdobných měření (ve stejné dny, ve stejnou denní dobu, na stejném místě.)

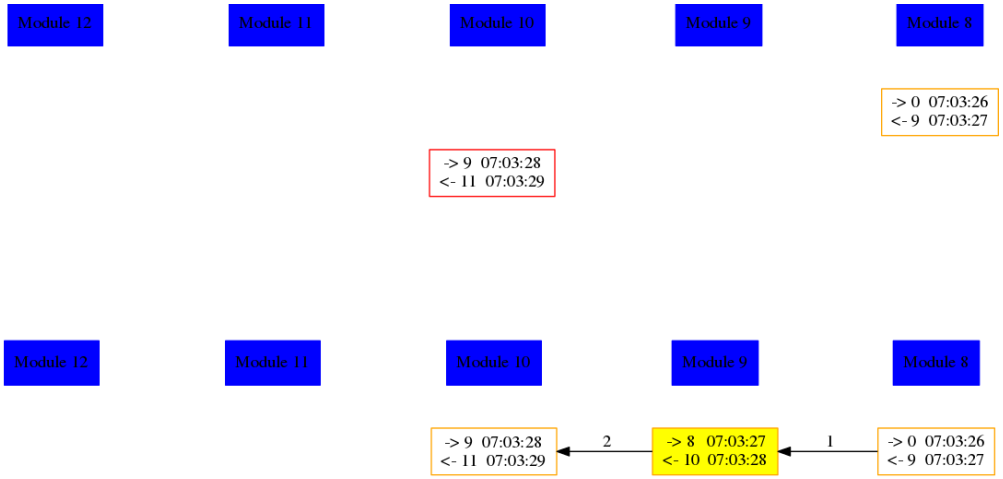
5.2.2 Chybějící záznamy o zákaznících

Jelikož podstata této chyby je podobná výpadku, budou se podobat i jejich řešení. Pokud totiž proběhne celá úvodní fáze slučování zákazníků, a někteří zůstanou nezařazeni do žádné sekce, znamená to, že v některém ze sousedních modulů chybí data. Tedy, že některý z modulů měl "dočasný výpadek."

Použitím upravené výpadkové modifikace metody slučování zákazníků je

tedy možné sloučit i tyto zákazníky, přestože ztráta dat nebyla způsobena technickým selháním záznamového zařízení.

Obrázek 5.1: Grafické znázornění doplnění chybějícího záznamu o zákazníkovi na základě dat ze sousedních modulů.



5.2.3 Vícenásobný odchod

Na základě dat ze sousedních modulů se vyhodnotí, do kterého modulu zákazník spíše odešel, a s ním budou data spojena.

Pokud některý sousední modul obsahuje časově odpovídající informace o příchodu zákazníka z daného sousedního modulu, bude spojen s ním. Pokud tyto informace neobsahuje, bude proveden kvalifikovaný odhad založený na běžně očekávaném chování zákazníka. (Například je pravděpodobnější odchod do jiného modulu, než ze kterého zákazník přišel.)

Metodu slučování dat o zákaznících je v tomto případě nutné upravit tak, že v případě, že modulový záznam s vícenásobným odchodem bude na konci sekvence, bude se posuzovat jako vhodný kandidát na předchůdce modulových záznamů z obou sousedních modulů.

5.2.4 Nesynchronizovanost

Při spojování zákazníků napříč jednotlivými moduly se poskytuje určitá tolerance rozdílu časových značek, při jejímž dodržení jsou záznamy ze sousedních modulů považovány za záznamy jednoho zákazníka.

V ideálním případě by byly modulové záznamy spojovány na základě shodné časové známky. Nicméně jak již bylo uvedeno, časové známky napříč moduly nejsou shodné. Při slučování modulových záznamů je tedy nutné projevit jistou benevolenci při výkladu výrazu "shoda časových značek."

Úprava metody slučování zákazníků pro odstranění této chyby se projeví změnou části jediné podmínky. A to tak, že výraz

```
r.origin\_time == c.destination\_time
```

se změní na

```
abs(r.origin\_time - c.destination\_time) < TOLERANCE
```

Tato změna zajistí, že podmínku časové shody budou splňovat i záznamy, jejichž čas sice není naprosto shodný, nicméně splňují stanovenou toleranci. Míra tolerance musí být stanovena s ohledem na (ne)spolehlivost časových značek.

5.2.5 Slepá místa

Tyto chyby nejsou příliš časté, a navíc jsou svými důsledky podobné některým jiným chybám. Nezaznamenaný průchod modulem se z tohoto důvodu řeší stejně jako výpadek nebo chybějící záznam o zákazníkovi.

Odchod mimo zorné pole se projevuje „neobvyklým“ chováním (zákazník opakovaně odchází z a vchází do modulu). Nicméně toto chování nemusí být vždy důsledkem chyby, zákazník se tak může skutečně chovat (například z nerozhodnosti). V datech, která byla při tvorbě práce použita, však nebyl dostatek takovýchto případů na to, aby bylo možné určit, kolik z nich je důsledkem chyb.

5.2.6 Předčasné vrácení

Jediná situace, kdy je chyba v interpretaci akcí manipulace se zbožím detekovatelná, je právě předčasné vrácení. Přestože situace, kdy zákazník vrátí zboží do regálu dříve než si z něj nějaké vezme může teoreticky nastat (už v sekci dříve byl, a nyní se do ní vrátil, případně pokud vrací zboží jinam, než kam patří), je to mnohem méně časté, než chybné vyhodnocení.

Tyto situace jsou tedy vždy považovány za chybu. Náprava v tomto případě však nevyžaduje zásah do slučování zákazníků. Je možné ji provést již po načtení dat, a to prostým převedením akce 6 na akci 4.

5.3 Rekonstrukce modulových záznamů

Jelikož data nejsou dokonalá, mohou v důsledku výpadků či chybné konverze hloubkových map chybět některé modulové záznamy o zákaznících. I v takových případech však chceme modulové záznamy týkající se téhož zákazníka spojit do jedné sekvence.

Pokud k tomu využijeme některá z modifikací metody slučování dat o zákaznících, některé z nalezených sekvencí nemusejí splňovat některé z uvedených podmínek. Jelikož jejich dodržení je však důležité pro budoucí analýzu dat, je nutné chybějící moduly doplnit, tedy zrekonstruovat.

Při rekonstrukci těchto modulů je nutné vycházet z dat ze sousedních modulů. Pokud máme časově podobná data ze dvou nesousedních modulů, můžeme doplnit data ze všech modulů mezi nimi. Při určování toho, co je ještě přijatelná časová podobnost, je nutné vzít do úvahy počet rekonstruovaných modulů. Jako nejschůdnější řešení se nabízí zvolit nějakou časovou konstantu a tu násobit počtem modulů. Pokud bude časový rozdíl mezi moduly vyšší, není možné je prohlásit za součást téže sekvence, neboť zde je již celkem reálná možnost, že každý modulový záznam zachycuje jiného zákazníka.

Stejně se dá postupovat u rekonstrukce příchodu do sekce či odchodu z ní. Jen s tou výjimkou, že zde nebude relevantní čas. Pokud zákazníkovi chybí v sekvenci modulový záznam o příchodu z krajního modulu, prostě se doplní.

Při všech rekonstrukcích je klíčové doplnit odkud zákazník přišel, kam pokračoval a časové známky. Všechny tyto údaje lze však určit z již použitých sousedních dat.

5.3.1 Rekonstrukce dalších akcí

Pro statistické účely je však rovněž důležité doplnit i akce 3; 4; 5 a 6, jinak by to negativně ovlivnilo jejich vnímání chování zákazníků. Jelikož tyto akce nelze doplnit z okolních dat, je na to nutné použít jiné metody.

Z okolních dat není možné ani odhadnout jakým způsobem se zákazník choval uvnitř modulu, z nějž chybí data. To však neznamená, že to nejde odhadnout z jiných dat. Pokud máme k dispozici větší množinu dat (data z téhož modulu za více dnů), máme k dispozici rovněž možnost rekonstruovat tato data na základě dat z obdobných časových úseků z jiných dnů.

Zde se nabízejí dva přístupy. 1) Zjistit si pravděpodobnost určité akce či skupiny akcí a náhodně určit, zda ji danému zákazníkovi přiřknout. 2) Zjistit si pravděpodobnost určité akce a přiřknout ji vždy tomu modulovému záznamu, který byl rekonstruován jako tolikátý v pořadí, že dosáhl převrácené hodnoty oné pravděpodobnosti.

Rovněž je nutné dávat si pozor, aby zvolená období byla statisticky relevantní. Např. při doplňování akcí u modulu ze soboty je vhodné určovat pravděpodobnost pouze z víkendových modulů.

5.4 Neodstranitelné chyby

Neodstranitelné chyby jsou takové, které není možné s dostupnými daty vyřešit, či někdy dokonce ani detekovat.

5.4.1 Manipulace se zbožím

S výjimkou situace předčasného vrácení, není možné nijak poznat, kdy jsou akce manipulace se zbožím vyhodnocovány správně, a kdy chybně.

Tyto případy jsou ignorovány, neboť je není možné s dostupnými prostředky vyřešit. K jejich úplnému odstranění je nutné zlepšit algoritmus převádějící hloubkové mapy na datovou tabulku.

5.5 Shrnutí

Data v datové tabulce není v žádném případě možné považovat za bezpodmínečně správná. Spíše naopak. Některé chyby je možné odstranit za použití ostatních dat, některé však vyžadují zásah do algoritmu, který tabulku vytváří, či dokonce do měřícího hardwaru.

Ani tato práce si neklade za cíl odstranit všechny chyby. Soustředí se pouze na ty z nich, které je možné - do určité míry - odstranit z dat obsažených v datové tabulce samé.

Kapitola 6

Implementace

Nyní již máme k dispozici teoretický popis problematiky, i nástin možných řešení. To, zda tato řešení opravdu fungují, ověříme jejich implementací.

Tato část se popisuje některá rozhodnutí, která bylo nutno učinit před zahájením implementace popsaných metod, a rovněž implementaci samotnou.

6.1 Programovací jazyk

Bez programovacího jazyka se dnes při programování lze jen velmi těžko obejít. A jeho volba má významný dopad na vlastní podobu projektu. Proto je důležité zvolit si takový jazyk, ve kterém bude možné projekt kvalitně vytvořit, a zároveň ideálně takový, který bych se nemusel učit od píky.

Výhodou tohoto projektu je, že u něj není na prvním místě vyžadována efektivita. Program, který bude výsledkem této práce bude používán pouze po skončení mnohadenního cyklu měření, a to až v době, kdy čas nebude relevantní. To, zda tedy bude program opravovat chyby v záznamech z jednoho dne 10 sekund, nebo 10 minut, není příliš důležité.

Z tohoto důvodu jsem tedy nebyl nucen volit rychlé a efektivní jazyky, které svoji efektivitu vykupují nízkoúrovňovostí, a mohl jsem klidně zvolit svůj oblíbený Python[8], který mi umožňuje psát snadný a přehledný kód.

6.2 Datové struktury a persistence

Jednou ze základních otázek, na které je rovněž důležité si odpovědět hned zpočátku, jsou použité datové struktury a ukládání dat.

V dnešní době jistě není pro nikoho překvapením, že jsem zvolil objektově-orientovaný přístup. Základní jednotky popsané v teoretické části jsem tedy realizoval ve formě následujících tříd:

Action Třída reprezentující jednotlivou akci tak jak je popsaná v teoretické části. Obsahuje 10 základních hodnot získaných z řádku datové tabulky. (Datová tabulka obsahuje hodnot 11. Třída Action neobsahuje hodnotu Date, neboť tato je již součástí hodnoty Time).

SectorVisit Třída reprezentující modulový záznam. Obsahuje ID zákazníka, časy a místa příchodu a odchodu, všechny další akce a informaci o tom, jestli byl záznam získán z datové tabulky, nebo rekonstruován.

Customer Třída reprezentující zákazníka jako sekvenci modulových záznamů. Obsahuje akorát unikátní ID a uspořádaný List modulových záznamů.

Module Třída reprezentující modul jako modifikovanou datovou tabulku. Obsahuje modulové záznamy tříděné na bezproblémové a chybné (např. vícenásobný odchod.)

Persistenci (tedy ukládání dat mezi relacemi) jsem zpočátku řešil pomocí Pythonovského integrovaného modulu pickle. Toto řešení se však brzy ukázalo jako málo pružné, a pro potřeby tohoto projektu nedostatečné, takže jsem ho brzy nahradil SQLite databází, která umožňuje mnohem efektivnější práci s daty a zejména efektivnější vyhledávání pro účely statistické rekonstrukce akcí.

Database Třída reprezentující databázi. Obsahuje metodu pro vytvoření databáze a metody pro ukládání objektů do databáze a čtení z ní.

6.3 Čtení dat

Před vlastní prací s daty je rovněž nutné vyřešit jejich čtení a ošetření. Vzhledem ke specifickému zdroji dat a úzkému okruhu uživatelů se dá rozumně předpokládat, že data budou více méně odpovídat požadované specifikaci. Není tedy nutné je testovat tak precizně, jako uživatelské vstupy.

Čtení probíhá přímo ze zdrojových souborů ve formátu xlsx. Po přečtení jsou data podrobena analýze, zda příslušné sloupce obsahují správné datové typy, jsou odstraněny duplicity a číselné hodnoty jsou uloženy jako čísla, nikoliv řetězce.

Parsers Třída obsahující metody pro čtení, ošetření a předběžné zpracování dat

6.4 Oprava dat

Nejpodstatnější činnost programu, totiž vlastní oprava dat, probíhá v rámci procesu slučování dat o zákaznících. Tento proces probíhá při vytváření instancí třídy Day, kde je vyjádřen čtyřmi jednoduchými funkcemi prováděnými v následujícím pořadí:

```
unify_modules()
fix_invalid_entries()
calculate_missing_elements()
fix_origins_and_destinations()
```

První tři z těchto funkcí pro vlastní slučování volají funkci `_join_paths`.

6.4.1 Funkce volající __join__paths

unify__modules Tato funkce prohledá všechny moduly příslušného dne, a všechny záznamy o zákaznících, které nejsou stiženy chybou vícenásobné akce odchodu, uloží do jednoho pole, na které pak zavolá funkci _join_paths.

fix__invalid__entries Tato funkce přidává do pole vytvořeného předchozí funkcí i zákazníky stižené chybou vícenásobné akce odchodu, a rovněž na ně volá funkci _join_paths. Důvodem rozdělení je to, aby nejprve byly slučováni pouze bezchybní zákazníci, a chyby byly řešeny až následně.

calculate__missing__elements Tato funkce opakovaně volá variantu funkce _join_paths, přičemž při každém volání jí dává větší volnost při dotváření chybějících údajů.

6.4.2 Funkce __join__paths

Toto je jedna z nejdůležitějších funkcí celého programu, neboť provádí slučování dat o zákaznících napříč moduly. Funkce vlastně vytváří z jednotlivých záznamů o zákaznících (uzlů) nespojitý graf[6] takovým způsobem, že jednotlivé uzly spojuje orientovanými hranami, které reprezentují přechod zákazníka z jednoho modulu do druhého.

Funkce je rovněž schopna doplnit chybějící uzel k tomu, aby mohla spojit dva existující uzly. Tato schopnost je omezena parametrem `creativity`, který udává maximální počet takto doplněných uzlů za sebou.

6.4.3 Funkce fix_origins_and_destinations

Tato funkce řeší problém zákazníků „jdoucích odnikud nikam," tedy zákazníků, u kterých chybí záznam o vstupu do sekce či o odchodu z ní. Tyto záznamy nejsou doplněny předchozí funkcí, protože nejsou potřeba ke spojování existujících záznamů.

Funkce fix_origins_and_destinations tedy doplňuje tyto chybějící údaje tak, aby každý záznam o zákazníkovi byl součástí nějaké řádně započaté i ukončené sekvence.

6.5 Grafické uživatelské rozhraní

Jelikož program není určen k používání širokou veřejností (a už vůbec ne laickou,) nehraje grafické uživatelské rozhraní (dále jen GUI) tak významnou roli jako u jiných aplikací. GUI zde neslouží jako prvotní prezentace programu, nemá uživatele „vodit za ručičku," ani nabízet přehršel funkcí. Jeho účelem je zde pouze jednoduchou cestou zpřístupnit dostupnou funkcionalitu.

Proto se GUI omezuje na jednoduché okno obsahující několik málo vstupních elementů (umožňujících uživateli zadat cestu ke zdrojovým souborům a adresář pro ukládání výstupů) a tlačítka umožnující spustit opravu chyb. GUI bylo vytvořeno za použití Pythonovského modulu tkinter42.

Nad rámec základní funkcionality ještě GUI umožňuje přístup k vizualizačním funkcím, jejichž účelem je zobrazení dat v uživatelsky přívětivé grafické podobě.

GUI Třída obsahující metody pro zobrazení GUI a metody propojující GUI s vlastními funkcemi aplikace.

6.6 Vizualizace

Jak již bylo naznačeno, program obsahuje i funkce, které pro samé odstraňování chyb nejsou nezbytné, nicméně byly významné při předběžné analýze dat a návrhu samotné vnitřní logiky hlavních funkcí, a i nyní mohou uživateli poskytnout přehlednější a lépe zpracovatelný pohled na data.

6.6.1 Graf modulových záznamů

Základním vizualizačním nástrojem je grafické znázornění modulových záznamů jednotlivých zákazníků. Části těchto grafů již byly k vidění v předchozích částech práce. Modulové záznamy téhož zákazníka představují uzly grafu a jsou zobrazeny na jednom řádku. Pořadí průchodu je naznačeno orientovanými hranami.

Uzly jsou opatřeny časy příchodů a odchodů a čísly modulů, z nichž a do nichž byly tyto akce vykonány. Hrany jsou číslovány vzestupně a udávají pořadí v němž zákazník přecházel mezi jednotlivými moduly.

Uzly mají rovněž různé barvy, které vyjadřují různé druhy dat:

Zelené ohraničení značí, že sekvence modulových záznamů je uzavřená (tedy začíná akcí příchodu do sekce a končí odchodem ze sekce.)

Oranžové ohraničení značí, že sekvence modulových záznamů je napůl uzavřená (tedy začíná akcí příchodu do sekce, nebo končí odchodem ze sekce.)

Červené ohraničení značí, že sekvence modulových záznamů je otevřená (tedy nezačíná akcí příchodu do sekce a nekončí odchodem ze sekce.)

Bílé pozadí značí, že modulový záznam byl importován z datové tabulky.

Fialové pozadí značí, že modulový záznam byl importován z datové tabulky a byly v něm opraveny chyby (například dvojí odchod).

Žluté pozadí značí, že modulový záznam byl rekonstruován z okolních dat a nemá předobraz v datové tabulce.

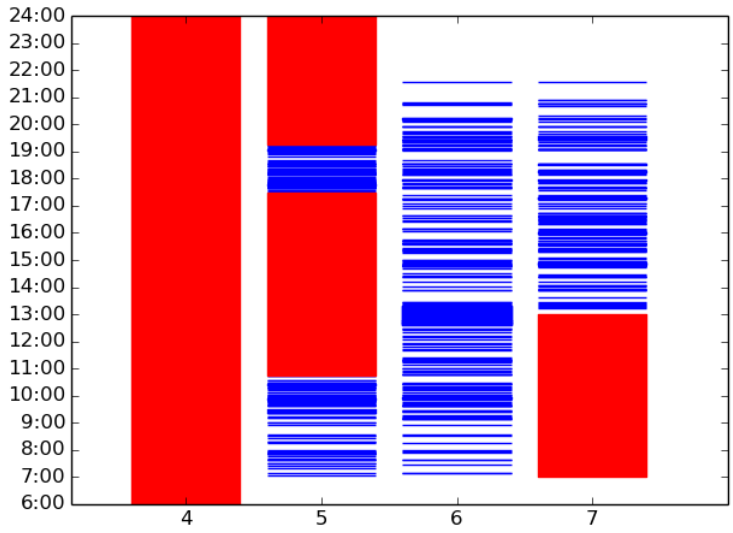
V příloze B je možné nalézt 4 příklady grafů modulových záznamů z různých fází procesu slučování zákazníků a odstraňování chyb.

6.6.2 Graf funkčnosti

V tomto grafu jsou zaznamenávány údaje o výpadcích jednotlivých modulů. Graf ukazuje pro jednotlivý den a sekci, které moduly byly kdy aktivní, a které měly hlášené výpadky. Porovnáním těchto informací je možné identifikovat nenahlášené výpadky, nebo naopak období, v nichž snímací zařízení fungovala, ale žádní zákazníci nepřicházeli.

Modře jsou zobrazena období, v nichž jsou v daných modulech aktivní nějací zákazníci. Červeně jsou zobrazena období výpadků.

Obrázek 6.1: Grafické znázornění doplnění chybějícího záznamu o zákazníkovi na základě dat ze sousedních modulů.



6.7 Složitost algoritmu

K tomu, aby byl nějaký algoritmus použitelný, nestačí, aby fungoval správně. Je rovněž nutné, aby byl schopný poskytnout výsledek v přijatelném čase, a za využití přijatelného množství prostředků. Meze toho, co je ještě možné považovat za přijatelné, jsou u různých aplikací různé, a záleží tedy na tom, k čemu bude algoritmus používán.

6.7.1 Časová složitost

Jak jsem již uvedl dříve, u tohoto algoritmu není předpokládáno jiné využití, než následné zpracovávání dat. Limitující podmínka tedy je, že algoritmus být schopný zpracovávat data rychleji, než jsou nová data generována.

Toto omezení však znamená jen to, že by zpracování dat za jeden den mělo trvat nejvýše 24 hodin. Při implementaci algoritmu jsem tak měl k dispozici poměrně velkou volnost, jelikož tento limit je velmi mírný.

6.7.1.1 Výpočetní doba

Program, který je přílohou této práce, implementuje algoritmus takovým způsobem, že na testovacích datech obsahujících záznamy z dvaceti sedmi dnů ve dvou kategoriích čítajících dohromady devět modulů trvá výpočet přibližně 24 minut. Data za jeden den jsou tedy zpracovávána v době kratší, než jedna minuta.

Počítač, na kterém byl program testován rozhodně nepředstavuje žádnou špičku z hlediska hardwaru:

Název	Acer Extensa 2508
Procesor	Intel Celeron M
Frekvence procesoru	2.16 GHz
Počet jader procesoru	2
Velikost operační paměti	4 GB
Grafická karta	Graphics Intel® HD Graphics

6.7.1.2 Asymptotická složitost

Kromě skutečné doby výpočtu je však relevantní i Asymptotická složitost[7] algoritmu. Výpočetně nejsložitější funkcí algoritmu je zde beze sporu funkce _join_paths, která provádí slučování informací o zákaznících.

Všechny ostatní funkce mají ve srovnání s touto funkcí složitost řádově nižší, a proto bude pro určení asymptotické složitosti stačit určit složitost této funkce.

Při určování složitosti funkce budu jako referenční jednotku uvažovat počet záznamů o zákaznících ve všech modulech v rámci téže kategorie v jednom dni, jelikož právě nad touto množinou funkce _join_paths operuje.

Funkce samotná má kvadratickou složitost, jelikož cyklicky prochází totéž pole, přičemž při každém průchodu z něj odstraní alespoň jeden prvek, nebo procházení ukončí. V nejhorším případě pole o n prvních projde n-krát.

Každá iterace tohoto cyklu však volá vnořenou funkcí, která má lineární složitost, jelikož prochází pole záznamů a hledá nejvhodnějšího následníka.

Celkem má tedy funkce _join_paths kubickou asymptotickou složitost.

6.8 Paměťová složitost

Program ke své činnosti nepotřebuje ukládat více, než zdrojová data a zrekonstruované záznamy. V nejhorším případě bude jeho paměťová složitost současně lineárně závislá na počtu záznamů a počtu modulů (záznamy je možné u každého zákazníka doplnit v každém modulu, i kdyby data obsahovala záznam pouze z jednoho modulu).

Kapitola 7

Hodnocení výsledků

Žádná práce by nebyla kompletní bez zhodnocení úspěšnosti. V tomto případě však hodnocení nebude vůbec jednoduché. Není totiž možné provádět jakékoliv automatizované testy ověřující správnost a úspěšnost implementace, jelikož k jejich vytvoření by bylo potřeba vyřešit tentýž problém, jaký řeší práce samotná.

7.1 Způsob ověřování výsledků

Při určování míry úspěšnosti při nápravě chyb v datech jsem tedy musel přistoupit k manuálnímu posuzování. Prakticky to probíhalo tak, že jsem procházel vizualizované výstupy mého programu záznam po záznamu a porovnával jsem je s hloubkovými mapami, na jejichž základě byla vytvořena zdrojová data.

Tento přístup s sebou pochopitelně přináší řadu nevýhod, jež mohou negativně ovlivnit vypovídající hodnotu takto získaných výsledků:

Malý rozsah dat - Vzhledem k náročnosti a zdlouhavosti této metody, jsem byl schopný takto ověřit správnost jen přibližně čtyř procent všech dat. Uváděná úspěšnost je tedy jen zobecněním této malé množiny.

Chyby při posuzování - Rovněž je možné, že některé mapy jsem vyhodnotil špatně. V důsledku toho se uváděná úspěšnost může od úspěšnosti skutečné lišit, a to v obou směrech.

I přes tato úskalí však zvolená metoda přináší relevantní výsledky na jejichž základě lze rozhodnout, zda mělo v této práci zvolené řešení vůbec smysl.

7.2 Zcela odstraněné chyby

Vliv některých chyb na data byl v průběhu procesu zcela eliminován. Jedná se především o:

- Duplicity
- Neočekávané znaky ve zdrojových souborech
- Špatně pojmenované soubory

Tyto chyby jsou jednoduché na identifikaci i odstranění, nevyžadují žádné doplňování chybějících informací a lze je bez problémů vyřešit ještě před slučováním informací o zákaznících. V opravených souborech tedy již nejsou ani duplicitní řádky, ani přebytečné znaky.

7.3 Částečně odstraněné chyby

V této části jsou zastoupeny ty chyby, jejichž náprava byla hlavním smyslem této práce. Žádná z těchto chyb nemohla být odstraněna úplně, protože data ze sousedních modulů, která byla k dispozici, neposkytují úplné informace, nýbrž jen jakási „vodítka“.

7.3.1 Chybějící záznamy o zákaznících

Porovnáním výstupů programu s hloubkovými mapami jsem zjistil, že když byly chybějící záznamy doplňovány mezi dvojicí již existujících záznamů, tak ve více než 80 % případů k nim existovala korespondující hloubková mapa, která však nebyla korektně převedena do datové tabulky.

Naproti tomu při doplňování záznamů spojujících existující záznamy vedoucí odnikud nikam s kraji sekce, je úspěšnost jen přibližně 40 %. Důvodem je z velké části pravděpodobně nepředvídatelné chování zákazníků (např. otočení se uprostřed sekce, a návrat stejným směrem, jakým přišli). Takto nízká úspěšnost však může rovněž naznačovat hlubší problémy snímacího zařízení, větší četnost chyb typu „slepá místa“ nebo existenci jiných chyb při konverzi hloubkových map na datové tabulky.

Nicméně i špatně doplnění zákazníci jsou pro účely statistické analýzy lepší, než zákazníci jdoucí odnikud nikam. Tímto doplněním nikdy nebyl vytvořen neexistující zákazník, takže na údaje o celkovém počtu zákazníků má použitá metoda výrazně pozitivní efekt, i když není dokonalá.

Spojováním údajů o zákaznících touto metodou se totiž uváděný počet zákazníků více přibližuje skutečnosti, než jak by tomu bylo bez doplňování chybějících informací.

7.3.2 Vícenásobný odchod

Porovnání s hloubkovými mapami ukazuje, že ve zhruba 75 % případů byl z množiny možných modulů, do nichž mohl zákazník odejít, zvolen ten správný.

Část byla určena nesprávně v důsledku nedostatku informací ze sousedních modulů (takže v podstatě bylo možné modul pouze tipnout) a část byla způsobena snahou minimalizovat počet doplňovaných záznamů preferencí odchodu mimo sekci, či do modulu, kde bude vyžadováno méně doplnění.

7.3.3 Výpadky

V případě této chyby je ověřování úspěšnosti ještě horší, než v předchozích případech, jelikož zde nejsou k dispozici ani příslušné hloubkové mapy, na jejichž základě by bylo možné výsledky ověřit.

Jediná informace, na níž je možné odhad úspěšnosti založit, je úspěšnost metody odstraňování chybějících záznamů o zákaznících. Povaha obou chyb je totiž podobná a implementace mezi těmito dvěma případy nerozlišuje, a tak je možné se důvodně domnívat, že úspěšnost v tomto případě bude obdobná.

7.4 Neodstranitelné chyby

Chyby v akcích manipulace se zbožím, jež není možné vůbec detekovat, nebyly odstraněny.

U nápravy chyby předčasného vrácení není možné posoudit její úspěšnost, jelikož nejsou k dispozici data, z nichž by bylo možné posoudit podíl legitimních vrácení zboží.

Z obdobného důvodu není možné úspěšně provádět statistické doplňování akcí manipulace se zbožím u doplněných záznamů. Doplněné akce by totiž byly stejně nesprávné, jako ty původní, jejichž míru správnosti není možné určit.

7.5 Shrnutí

Implementace dosahuje uspokojivých výsledků při odstraňování chyb v datech. Zlepšení úspěšnosti používaných metod brání nedostatečná kvalita dat používaných k rekonstrukci.

Pro efektivnější rekonstrukci by bylo potřeba pracovat nejen s datovou tabulkou, nýbrž i přímo s hloubkovými mapami. To by však vyžadovalo zlepšení algoritmu jejich konverze.

Závěr

Cílem této práce bylo analyzovat proces sledování chování zákazníků v prodejnách obchodních řetězců, popsat potenciální chyby, navrhnout jejich řešení a implementovat je.

Analýzou bylo zjištěno, že při používání zařízení pro záznam aktivit zákazníků a následném zpracování získaných údajů vznikají dva druhy chyb - chyby technické a chyby softwarové.

Na základě výsledků analýzy byly vybrány chyby, u nichž má smysl pokoušet se o jejich nápravu. Následně byly navrženy metody směřující k omezení vlivu těchto chyb na data či jejich úplnému odstranění.

Tyto metody byly následně implementovány a otestovány na reálných datech získaných při měření v jedné z prodejen. Výsledky byly následně analyzovány z hlediska úspěšnosti.

Některé chyby se podařilo zcela eliminovat, většinu ostatních alespoň výrazně napravit. Část chyb však byla způsobena chybnou konverzí výstupů záznamových zařízení do formátu, s nímž tato práce pracovala. Část těchto chyb nebylo možné z poskytnutých dat vůbec identifikovat, natož pak opravit.

Další navazující práce by se tedy mohla zaměřit za zlepšení konverzního algoritmu, neboť jeho nedostatky jsou příčinou těch nejobtížněji napravitelných chyb.

Literatura

Literatura

[1] *GfK Retail, Shopper & Regional Studies* [online]. [cit. 2016-05-16]. Dostupné z: http://incoma.cz/

[2] *Project Natal 101.* Seattlepi [online]. 2009 [cit. 2016-05-08]. Dostupné z: http://blog.seattlepi.com/digitaljoystick/2009/06/01/e3-2009-microsoft-at-e3-several-metric-tons-of-press-releaseapalloza/

[3] *Depth Buffers (Direct3D 9).* Microsoft [online]. [cit. 2016-05-08]. Dostupné z: https://msdn.microsoft.com/en-us/library/windows/desktop/bb219616(v=vs.85).aspx

[4] WALKENBACH, John. *Excel 2013 bible.* Indianapolis: Wiley, 2013. ISBN 978-1118490365. Dostupné z: http://ss64.com/bash/uniq.html

[5] *uniq man page.* Free Software Foundation [online]. 2010 [cit. 2016-05-08]. Dostupné z: http://linux.die.net/man/1/uniq

[6] MARTY, U.S.R. a J.A. BONDY. *Graph Theory with Applications.* Springer, 2008.

[7] TESAŘ, Karel a Martin MAREŠ. *Recepty z programátorské kuchařky: Složitost* [online]. [cit. 2016-05-16]. Dostupné z: https://ksp.mff.cuni.cz/tasks/25/cook1.html

[8] *Python* [online]. [cit. 2016-05-16]. Dostupné z: https://www.python.org/

[9] *Creating Excel files with Python and XlsxWriter* [online]. [cit. 2016-05-16]. Dostupné z:http://xlsxwriter.readthedocs.io/

[10] *Graphviz - Graph Visualization Software* [online]. [cit. 2016-05-16]. http://www.graphviz.org/

[11] *matplotlib* [online]. [cit. 2016-05-16]. Dostupné z:http://matplotlib.org/

Příloha **A**

Návod k použití programu

Přestože přiložený program je velmi jednoduchý na obsluhování, je na místě několik drobných upozornění spíše ve stylu FAQ, nežli manuálu.

A.1 Linux

Na začátek je potřeba uvést, že program je určen pro použití v unixovém systému. Přestože Python je multiplatformní jazyk, a je tedy možné je zkompilovat v libovolném z nejrozšířenějších operačních systémů, spustitelné verze jsou určeny výhradně pro Unix.

Pokud chcete, můžete si samozřejmě program zkompilovat i pod jiným operačním systémem. Některé funkce však vyžadují, abyste měli nainstalovány některé rozšiřující pythonovské moduly (xlsxwriter[9]) a programy třetích stran (Graphviz[10] a matplotlib[11]).

A.2 Pracovní adresář

Všechny výstupy z programu jsou umístěny ve složce output generované v adresáři src. Z tohoto důvodu není možné program spustit přímo z přiloženého CD, nýbrž je nutné zkopírovat ho do počítače.

Po zkopírování do počítače bude pravděpodobně nutné nastavit programu executable bit, aby jej bylo možné spustit.

Zdrojová data mohou být poskytnuta z disku.

A.3 Spuštění programu

A.3.1 Mód opravy chyb

Pro opravení chyb v datech je nutno poskytnout cestu ke složce s daty. Na disku jsou umístěny dvě složky s daty (data a test_data). Při vybírání složky si dejte pozor na to, abyste vybrali správnou složku. Před kliknutím na OK ve výběrové dialogu si zkontrolujte zobrazovanou cestu. Ta by měla končit složkou s daty.

Soubor s výpadky je nepovinný. Nicméně pokud ho chcete zadat, musíte zadat přímo soubor, nestačí složka.

Kliknutím na tlačítko start spustíte proces, během kterého dojde ke zpracování všech souborů ve zdrojové složce, a k uložení výstupů do složky output. Není možné volit, které dny se budou zpracovávat, a není možné proces přerušit jinak, než násilným ukončením aplikace. Zvažte tedy, jaké množství dat chcete mít ve složce s daty. (Složka test_data obsahuje data jen za 3 dny, zatímco složka data jich obsahuje mnohem více)

A.3.2 Generování vizualizací

Pokud chcete vygenerovat grafy popsané v sekci vizualizace, klikněte na show day-specific statistics, vyberte příslušný den, a klikněte na jedno ze dvou tlačítek, podle toho, jaký graf chcete. Graf bude rovněž uložen do složky output.

A.4 Light verze

Pokud se Vám nepodaří spustit plnou verzi z důvodu chybějících externích programů (nejčastěji matplotlib), je na disku k dispozici i verze postrádající některé doplňkové funkce (které nejsou nezbytné k dosažení cílů této práce), která nevyžaduje žádné externí programy.

A.5 Verze bez GUI

Pokud se Vám nepodaří spustit ani light verzi, je na disku obsažena i verze bez grafického uživatelského rozhraní určená k přímému použití z unixové příkazové řádky.

Tato verze je obsažena v adresáři cmd_line, a kromě vlastního spustitelného souboru je v něm i složka s testovacími daty. Program bere jako jediný argument právě složku se zdrojovými daty. Takže pro spuštění stačí zkopírovat celý adresář na disk, spustit příkazovou řádku, přidat souboru právo být spuštěn (chmod +x) a spustit program například příkazem:

```
./main test_data
```

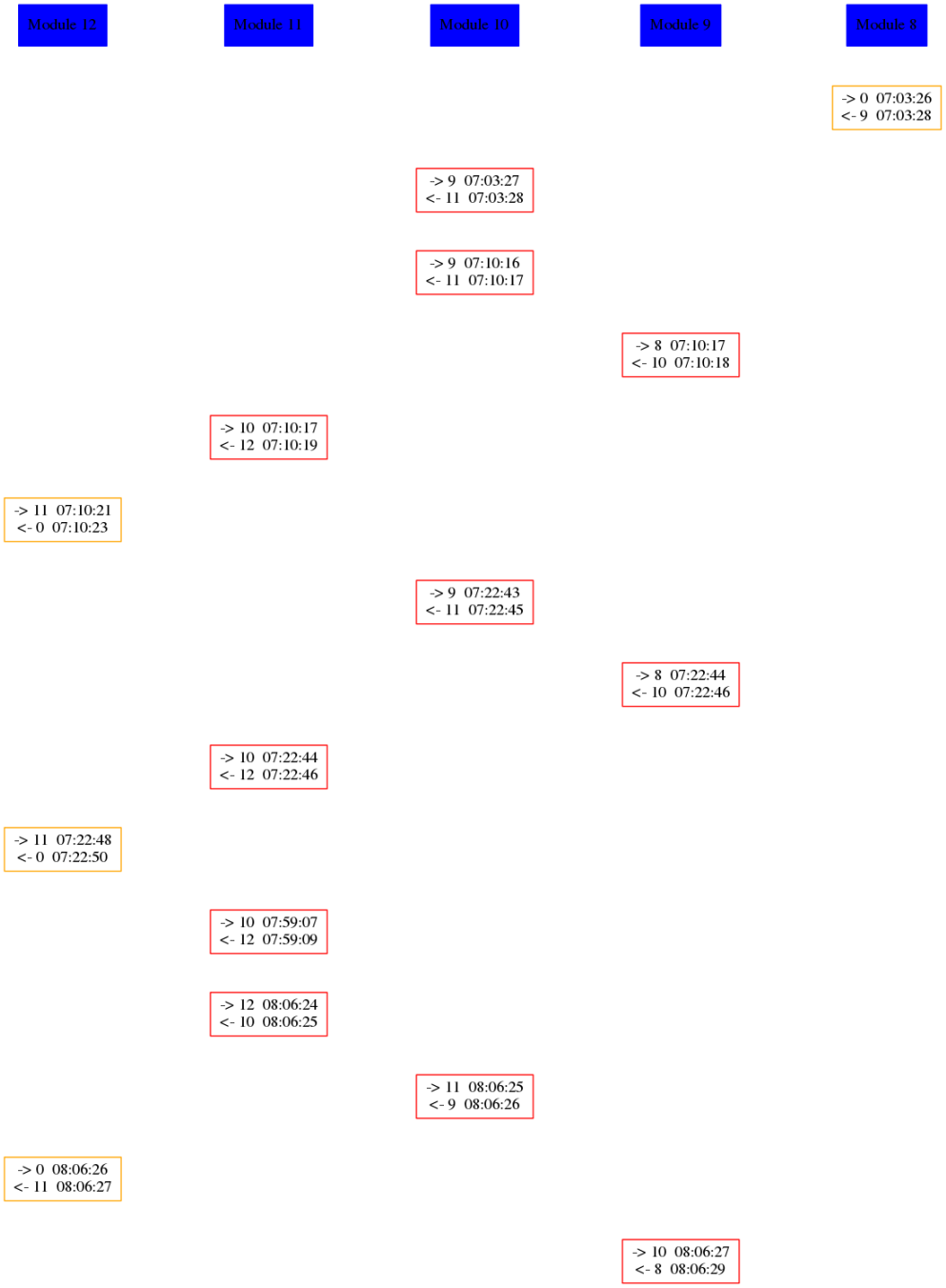
A.6 Upozornění

Přestože program je napsán tak, aby vzdoroval základním chybám uživatelských vstupů, je možné, že při práci s ním narazíte na nějakou neošetřenou chybu. V takovém případě, prosím, restartujte program, a zamyslete se nad tím, o co jste se pokoušeli. :)

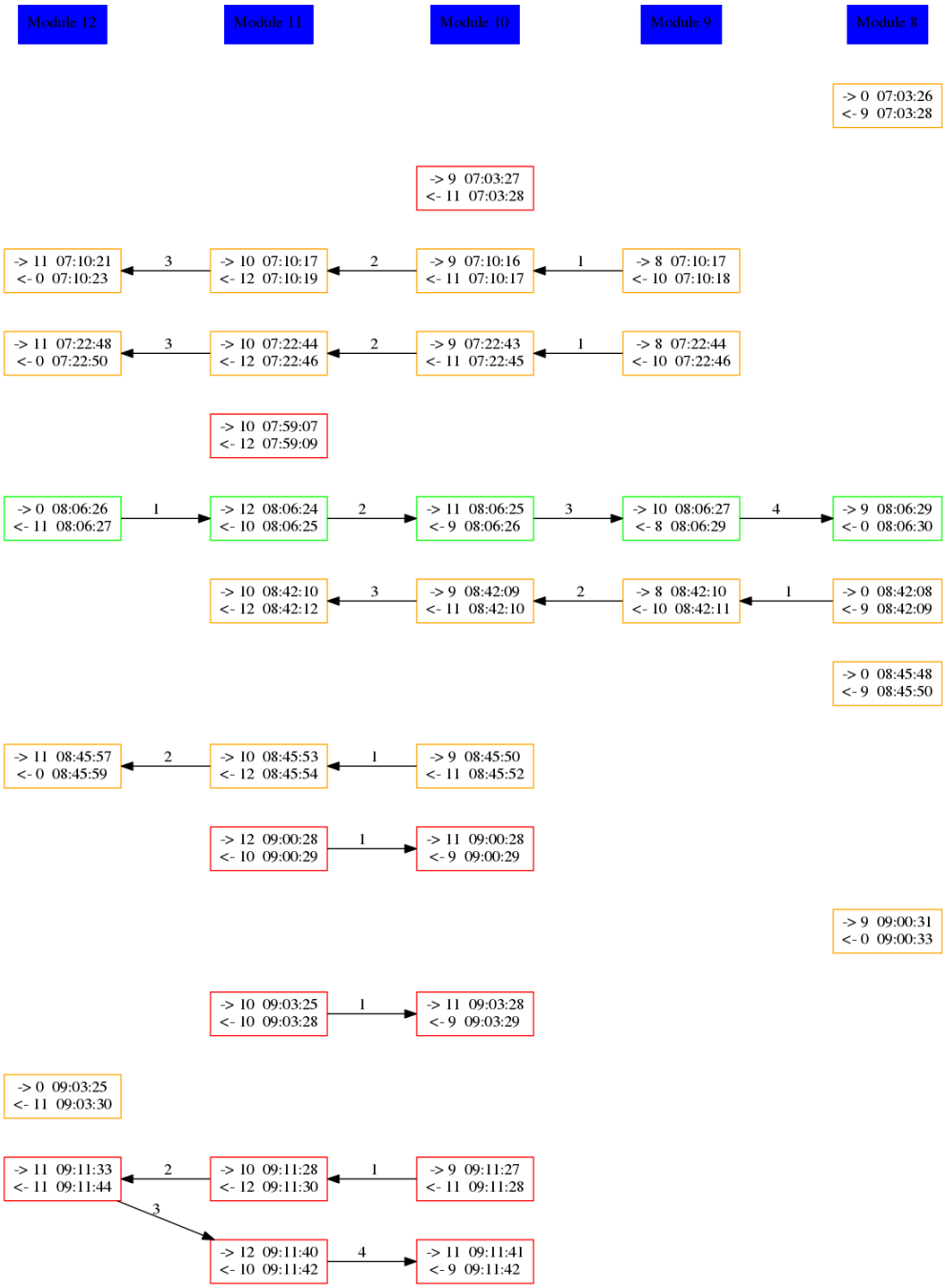
Příloha B

Grafy

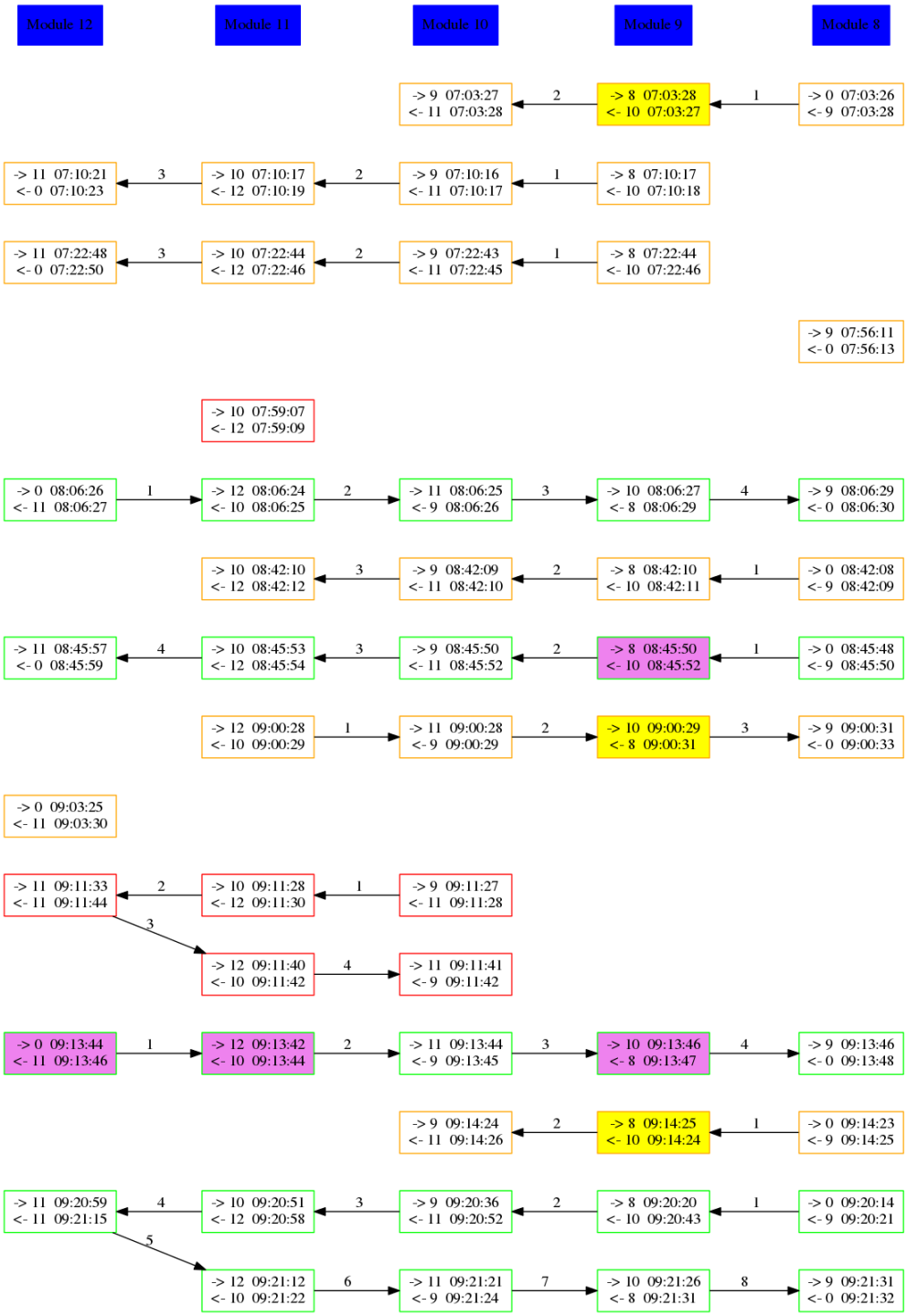
Obrázek B.1: Vizualizace zdrojových dat před opravou chyb a slučováním zákazníků.



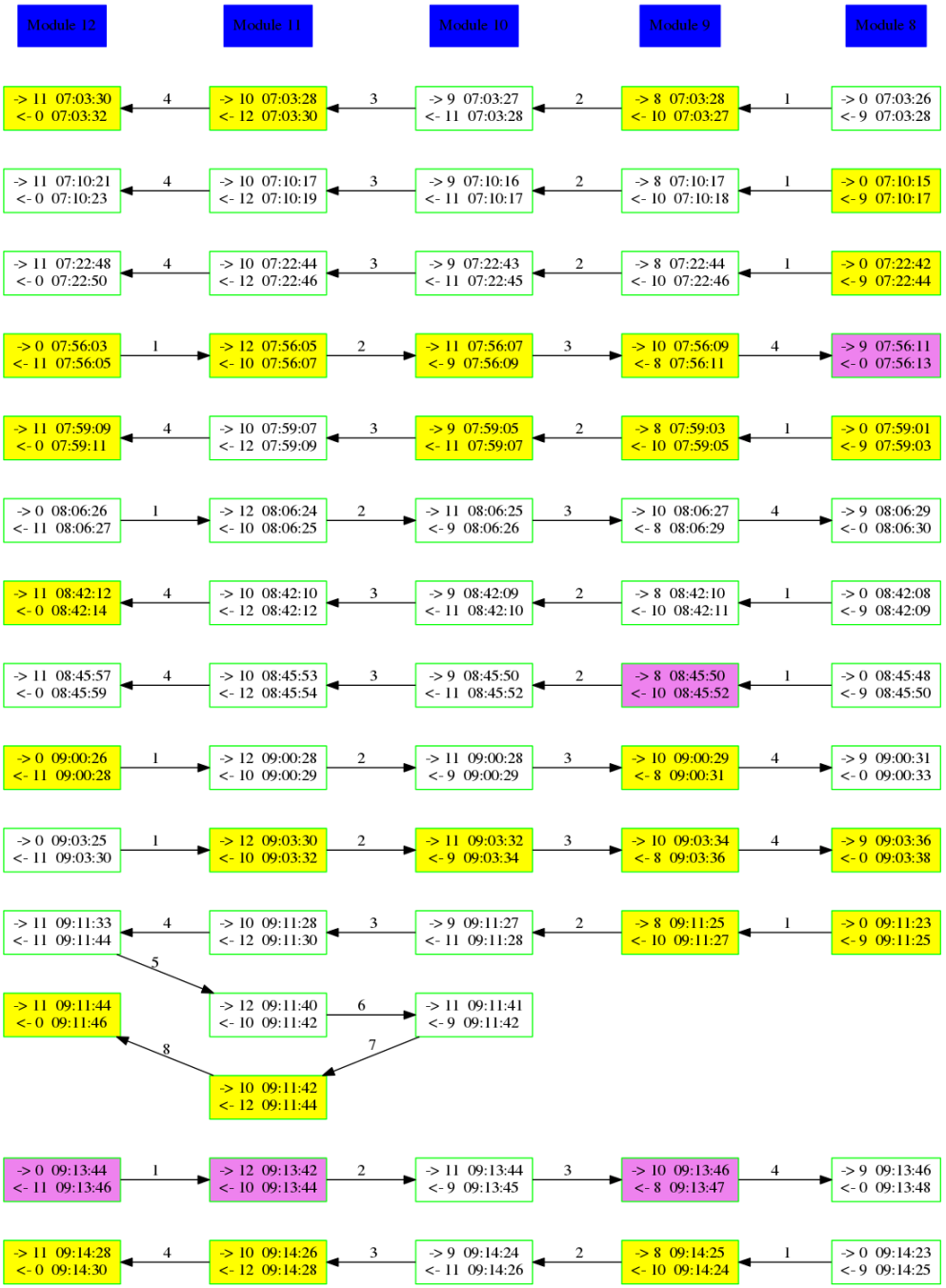
Obrázek B.2: Vizualizace zdrojových dat po provedení operace sloučení zákazníků.



Obrázek B.3: Vizualizace zdrojových dat po doplnění chybějících záznamů uvnitř sekce.



Obrázek B.4: Vizualizace zdrojových dat po doplnění chybějících záznamů na krajích sekce.



Příloha C

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `manual.pdf` ..... návod k použití přiloženého programu
- `exe` ..... adresář se spustitelnými formami implementace
  - `full` ..... úplná verze aplikace
  - `light` ..... verze nevyžadující mathplotlib
  - `cmd_line` ..... verze bez GUI a ostatních závislostí
- `data` ..... úplná data, které jsem měl k dispozici
- `test_data` ..... ukázková testovací data
- `src` ..... zdrojové kódy implementace
- `tex` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF