Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Diplomová práce

ETL nástroj pro konverzi OpenStreetMap dat do datové struktury nástroje TrafficModeller

Plzeň 2022 Vaněk Jakub

Místo této strany bude zadání práce.

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a výhradně s použitím citovaných pramenů. V knize jsou použity názvy programových produktů firem apod, které mohou být ochrannými známkami nebo registrovanými ochrannými známkami příslušných vlastníků.

V Plzni dne 22. června 2022

Vaněk Jakub

Poděkování

Tímto bych rád poděkoval panu Ing. Karlu Jedličkovi, Ph.D za ochotu při vedení diplomové práce a cenné rady s jejím vypracováním. Současně bych rád poděkoval své rodině a přátelům, kteří mě při vytváření této práce podpořili.

Abstract

ETL tool for conversion of OpenStreetMap data to Traffic Modeller tool data structure. Goal of this thesis is to create data for Traffic Modeller tool using OpenStreetMap data. This thesis describes and uses knowledge aboud geographic data and ETL tools. In the opening part geographic data, ETL process and tools relevant for this thesis are introduced. In the later part implementation design and final solution is described.

Abstrakt

Cílem této práce bude implementovat nástroj vykonávající konverzní algoritmus mezi daty s datovou strukturou OpenStreetMap a nástrojem Traffic-Modeller. V úvodní části jsou představena geografická data, ETL proces a datové struktury nástrojů relevantních pro tuto práci. Dále je popsána návrh implementace a finální řešení problému.

Obsah

1 Úvod

Cílem diplomové práce je vytvořit konverzní nástroj, který umožní automatizaci přípravy dat pro využití nástroje Traffic Modeller, který byl vyvinut ve spolupráci se Západočeskou Univerzitou v Plzni. Tento nástroj umožňuje pomocí svého API jednoduše a rychle simulovat a testovat různé scénáře dopravy na specifikovaném území.

K naplnění datové struktury tohoto nástroje mohou být použita volně dostupná data z OpenStreetMap. V takovém případě je pro naplnění datové struktury nástroje TrafficModeller potřeba získat data z OpenStreetMap, poté je transformovat do příslušné datové struktury nástroje TrafficModeller a následně je do něj nahrát, což umožní s daty dále pracovat a vytvářet simulace dopravních situací.

V úvodní části této práce jsou obecně popsána geografická data. Dále je představen proces ETL se zaměřením na tento typ data. Následně jsou popsány datové struktury vstupních dat OpenStreetMap a očekávané datové struktury dat pro nástroj Traffic Modeller. Další část se detailně věnuje popisu konverze mezi datovými strukturami, ze které vychází část popisující implementovaný nástroj vykonávající konverzní algoritmus.

V závěru práce jsou popsány a zhodnoceny dosažené výsledky. Součástí diskuze jsou pak návrhy na vylepšení konverzního algoritmu, datové struktury a způsob práce s nástrojem Traffic Modeller.

2 Geografická data využívaná v práci

Diplomová práce se zabývá prací s geografickými daty. V této kapitole jsou data definována, jsou zde popsány modely, které se využívají k jejich reprezentaci a jejich datové struktury. Dále jsou v kapitole popsány přístupy k harmonizaci dat.

http://geomatika.kma.zcu.cz/studium/sgg/Materialy/Vlastnosti_prostorovych_

2.1 Definice

http://geomatika.kma.zcu.cz/studium/sgg/Materialy/data.pdf

Data popisující prostorové informace nazýváme geografická data. Taková data se skládají ze dvou složek **https://kgm.zcu.cz/studium/ugi/elearning/index1.htm**. První složkou je tzv. popisná složka. Tato část dat popisuje vlastnosti reálných objektů. Druhou složkou je pak složka prostorová obsahující prostorové určení objektu. Tato data v sobě uchovávají informace o poloze, tvaru a vztazích mezi jevy reálného světla vyjádřené zpravidla formou souřadnic. Jsou to jakákoliv data obsahující formální polohovou referenci vztaženou k zemskému povrchu.

2.2 Reprezentace geodat

Geografická data obsahují tři základní typy informací:

- prostorová informace - vyjadřuje polohu objektu a jeho tvar
- popisná informace - další vlastnosti popisující objekt např.: název, adresa, teplota, tloušťka drátu apod.
- časová informace - pokud je použita časová informace, přidává systému dynamickou vlastnost, zaznamenává tedy čas změn

Tyto informace je možné reprezentovat analogově na mapě i digitálně v informačním systému.

Prostorová složka geografických dat je reprezentována pomocí mapových elementů. Ukládá se jejich umístění a tvar v prostoru, tedy geometrie. Popisná informace se na mapách reprezentuje pomocí kartografických vyjadřovacích prostředků (barvy, typy a tvary symbolů a čar, nápisy...). Můžeme říct, že charakteristika objektu (tj. jeho popisná informace) je na mapě reprezentována různým grafickým vyjádřením v závislosti na hodnotě atributu, např: je-li komunikace dálnicí, je vyjádřena tlustou žlutou čarou, je-li silnicí 1. třídy, je vyjádřena tlustou červenou čarou, která se postupně ztenčuje v závislosti na snižování třídy silnice.

https://www.natur.cuni.cz/geografie/geoinformatika-kartografie/kestazeni/projekty/moderni-geoinformacni-metody-ve-vyuce-gis-a-kartografie/prostor data/

V digitální formě jsou geografická data obvykle reprezentována rastrovou nebo vektorovou formou. Rastrová reprezentace (více např. viz **https://kgm.zcu.cz/studium/u** není v práci nijak využívána, není v ní tedy dále popisována. Nástroje používané v diplomové práci využívají model vektorový, který je dále popsán.

2.2.1 Reprezentace pomocí vektorů

Vektorová reprezentace se zaměřuje na popis jednotlivých geografických prvků pomocí úseků křivek s danou velikostí a směrem - tedy vektorů. K reprezentaci všech elementů jsou využity tři základní typy vektorových primitiv, pomocí kterých lze sestavit složitější mapové elementy. Jsou jimi bod, linie a plocha.

- bod - reprezentují bezrozměrné objekty, nebo objekty jejichž rozměr je tak malý, že nemá smysl jej reprezentovat plochou, např. sloup veřejného osvětlení
- linie - reprezentuje objekty jako řeky, silnice, potrubí, vedení, tedy objekty tak úzké, že je v měřítku mapy není vhodné reprezentovat plochami nebo také objekty, které nemají definovanou šířku (vrstevnice, ...).
- plocha - reprezentuje objekty, jejichž hranice uzavírá nějakou homogenní oblast (například jezera, lesy, zastavěná plocha, ...).

Prostorová složka jednotlivých mapových elementů je vyjádřena pomocí těchto primitiv. Jsou definovány svými souřadnicemi v prostoru. Linie (Line) je sestavena sekvencí sousedících úseček. Tyto úsečky jsou spojovány v mezilehlých bodech (vertex) a mají počáteční a konečný uzel (node). Polygony

jsou pak uzavřené linie, či uzavřený řetězech linií. Takový element má první a poslední uzel identický.

K odvození prostorových vztahů však počítači nestačí jednotlivé elementy. Je třeba zavádět topologii. Topologie je způsob vyjádření vztahu mezi jednotlivými elementy **https://kgm.zcu.cz/studium/ugi/elearning/index1.htm**. Výhodami topologie je například efektivnější uložení dat, či efektivnější práci s daty pro algoritmy, které nevyužívají prostorová data, ale pouze vztahy mezi jednotlivými elementy.

Popisná složka dat je vyjádřena skupinou atributů každého záznamu. Atributy jednotlivých elementů silně závisí na elementu, který popisují. Tyto atributy mohou nabývat nejrůznějších hodnot a slouží k popisu vlastností elementu. Název atributu by měl odrážet vlastnost kterou popisuje. Skupiny atributů mohou být libovolně podrobné a do detailu popisovat skutečné vlastnosti elementů.

2.2.2 Struktury vektorových datových modelů

Existuje několik datových modelů, ve kterých jsou geografické objekty uloženy. Liší se jak ve složitosti struktury, tak i v možnostech využívání topologických vztahů.

Špagetový model

Tento model patří mezi nejjednodušší. Každý mapový objekt je v něm zanesen vlastním záznamem. Každý záznam tedy obsahuje řetězec se seznamem souřadnic bodů, které jej tvoří. Největší nevýhodou tohoto modelu je absence informací o vztazích mezi jednotlivými objekty. Jedná se tedy pouze o seznam objektů se souřadnicemi, který nemá žádnou logickou strukturu.

Základní topologický model

Topologický model je jedním z nejpoužívanějších modelů uchovávající prostorové vztahy. V tomto modelu každá linie začíná a končí v bodě nazývaném uzel (node). V místě protnutí dvou linií vždy leží uzel. Každá část linie je uložena s odkazem na uzly a ty jsou uloženy jako soubor souřadnic x,y. Ve struktuře také uloženy identifikátory označující pravý a levý polygon vzhledem k linii. Tímto způsobem jsou zachovány základní prostorové vztahy použitelné pro analýzy. Navíc tato topologická informace umožňuje aby body, linie a polygony byly uloženy v neredundantní podobě. Tímto způsobem jsou v topologickém modelu uloženy i vztahy mezi jednotlivými objekty

Nevýhodou topologického modelu je opět neuspořádanost dat. Při vyhledávání jednotlivých objektů v různých analýzách může nastat situace, kdy se budou soubory s daty procházet celé a několikrát.

Hierarchický model

Tento model data ukládá v podobě logické hierarchie. Využívá samostatné tabulky pro uzly, linie a polygony. Linie pouze referencují záznamy z tabulky uzlů. Polygony pak referencují záznamy z tabulky jednotlivých linií. Hierarchický vektorový model nabízí výhody oproti topologickému modelu především při vyhledávání a manipulaci. Rozdělení polygonů, linií a bodů do různých souborů (nebo tabulek) umožní při vyhledávání použít pouze část datových struktur a tím urychluje práci.

Datové modely použité v práci

V práci jsou využívány základní liniové topologické vazby (konektivita), spolu s dalšími specifickými omezeními (směrovost, možnosti průchodu uzlem), které jsou ovšem ve zdrojových datech kódovány jinak, než v cílových. Podrobný popis datových struktur jednotlivých modelů popisuje kapitola 4.

2.3 Souřadnicové systémy

https://www.natur.cuni.cz/geografie/geoinformatika-kartografie/ke-stazeni/projekty/moderni-geoinformacni-metody-ve-vyuce-gis-kartografie-a-dpz/souradnicove-systemy/

Prostorová data se od těch neprostorových liší právě záznamem o své poloze. K určení této polohy využíváme souřadnicové systémy. Souřadnice v nich tvoří matematický zápis polohy objektu referencované ke konkrétnímu místu na, nad nebo pod zemským povrchem. Díky tomuto matematickému zápisu mohou souřadnice sloužit k různým výpočtům a analýzám. Polohu každého bodu na území je tedy možné jednoznačně určit právě pomocí souřadnic. V dnešní době existuje velké množství různých souřadnicových systémů. Některé jsou globální (WGS-84) a některé se využívají pouze lokální území (na území ČR S-JTSK). Nástroje v diplomové práci využívají globální souřadnicový systém *WGS-84*, který je dále popsán. Další souřadnicové systémy a jejich popis je dostupný na `http://epsg.io`

WGS-84

World Geodetic System 1984 (zkratka WGS-84) je světově uznávaný geodetický standard vydaný ministerstvem obrany USA v roce 1984. Jedná se o geocentrický pravoúhlý pravotočivý systém pevně spojený se Zemí. Definuje souřadnicový systém a referenční elipsoid WGS84 pro geodézii a navigaci. Odchylky od referenčního elipsoidu pak popisují geoid EGM84. V roce 1996 byl rozšířen o zpřesněnou definici geoidu EGM96. Byl vytvořen na základě měření pozemních stanic družicového polohového systému TRANSIT a nahrazuje dřívější systémy WGS60, WGS66 a WGS72. Tento systém je spojen s reálnou Zemí prostřednictvím souboru přesných souřadnic WGS84 pozemních stanic kontrolního segmentu GPS.

Polohu určíme pomocí zeměpisné délky, šířky a výšky. Šírka nabývá $0° - 90°$ na sever od rovníku a $0° - 90°$ na jih od rovníku. Délka pak nabývá hodnot $0° - 180°$ na západ od nultého poledníku a $0° - 180°$ na východ od nultého poledníku. Nultým poledníkem ve WGS84 je IERS Reference Meridian. Leží 5.31 úhlových vteřin východně od Prime meridian (Greenwich).

http://gnss.be/systems_tutorial.php

2.4 Geografické informační systémy

Geografická data jsou využívána geografickými informačními systémy. Informační systém je soubor hardware a software na získávání, uchovávání, spojování a vyhodnocování informací. Informační systém se skládá ze zařízení na zpracování dat, systému báze dat a vyhodnocovacích programů **(Clause a Schvill 1991)**. Geografický informační systém (GIS) je tedy informační systém pracující oproti klasickým informačním systémům navíc i s prostorovou složkou dat. Také lze říci, že je výkonným nástrojem geověd, tedy že metody těchto věd umožňuje efektivně implementovat v počítačovém prostředí. GIS slouží k analýze a modelování existujícího světa.

Příklady využití GIS:

- mapové služby
- obchod - analýza lokalit nových provozoven na základě demografických údajů, síťové analýzy rozvozu zboží ...
- ochrana proti pohromám - modely povodní, směrování záchranných prostředků
- distribuční společnosti - databáze kabelů, plynovodu, analýzy sítí, směrování v sítích

- životní prostředí - chování ekosystémů, modely znečišťování ovzduší
- státní správa, městské úřady - dopravní analýzy, volby, sčítání lidu, evidence
- školy - výuka geověd

Z pohledu definice GIS je využitý nástroj OSM (viz kapitola 4.2) geografický informační systém, který poskytuje mapové služby. Nástroj TraMod (viz kapitola 4.1) je pak informační systém poskytující síťové analýzy nad dopravní sítí.

3 Principy ETL

Zkratka ETL reprezentuje populární třífázový proces, při kterém jsou data z jednoho či více heterogenních zdrojů nahrána do datového skladu. Těmito třemi fázemi jsou fáze extrakce (z angl. *extraction*), transformace (z angl. *transform*) a zápis dat (z angl. *load*). Běžným označením pro prostředky ETL je rovněž datová pumpa.

Datový sklad je jednou ze základních komponent BI (*Business Intelligence*). William Inmon datový sklad definuje takto: "*A data warehouse is a subject-oriented, integrated, nonvolatile and time-variant collection of data in support of management's decision*". Jedná se o zvláštní typ databáze, která je používána pro datové analýzy nad rozsáhlými soubory dat. Inmon ve své definici používá čtyři důležité charakteristiky takové databáze:

- *subject-oriented* - orientovaný na subjekt - Datový sklad je orientovaný na subjekt, protože poskytuje informace o konkrétním subjektu namísto probíhajících operací organizace. Těmito subjekty mohou být zákanící, dodavatelé, prodej, výnosy apod. Datový sklad se nezaměřuje na probíhající operace, ale zaměřuje se na modelování a analýzu dat za účelem rozhodování.

- *integrated* - integrovaný - Ze všech charakteristik je právě tato tou nejdůležitější. Data jsou do datového skladu integrována z více různých zdrojů, které mohou mít rozdílnou strukturu, názvosloví, jednotky apod.. Taková data tedy musí být očištěna a transformována tak, aby se do datového skladu nahrála v jedné konzistentní podobě a byla tak umožněna jejich analýza.

- *non-volatile* - stálost - Tato charakteristika značí skutečnost, že data v datovém skladu jsou neměnná. Do datového skladu data pouze nahráváme a následně k nim přistupujeme, nikdy je však neaktualizujeme. Data jsou do datového skladu nahrávána v podobě statického záznamu, který reflektuje stav v daném čase. Pokud se tedy takový stav změní, není v databázi aktualizován, ale je nahrán nový záznam opět reflektující stav v novém čase a starý záznam je pro analytické účely v datovém skladu zachován.

- *time-variant* - časová variabilita - Data jsou do datového skladu nahrána tak, že reflektují stav v přesně daném čase. Tyto stavy jsou

tedy v datovém skladu zachovány a tím je umožněno získat přesný stav systému v jakémkoliv okamžiku.

V současném obchodním světě se různé společnosti potýkají s roustoucím množstvím sbíraných a uchovávaných dat a s potřebou těmto datům co nejlépe porozumět. Tato data mohou být ve společnostech používána k optimalizaci firemních procesů, sledování účinnosti firemních strategií, objevování nových nevyužitých příležitostí a mnohému dalšímu. Využití dat k rozhodovacím procesům ve společnosti nazýváme *Business intelligence*. Business intelligence chápeme jako soubor technologií a procesů, které uživatelům umožňují přístup k datům a analýzu dat za účelem podpory rozhodování [Howson].

K analýze dat je tedy nejdříve třeba vytvořit datový sklad, který je tvořen právě procesem *ETL* a je hlavní komponentou BI. V následujících částech budou popsány jednotlivé fáze procesu ETL.

3.1 Extrakce dat

Prvním procesem, který je používán při výstavbě datového skladbu je proces zvaný *extrakce*. Poté co určíme cíl datového skladu a stanovíme jeho strukturu, je třeba identifikovat zdrojové systémy, ze kterých budou data do datového skladu extrahována. Tyto zdrojové systémy se od sebe navzájem liší. Mohou se lišit například ve své struktuře, či formátu uložení. Běžnými formáty, ve kterých jsou data uložena mohou být například relační databáze, formát XML či JSON, ale může se jednat o jakékoliv jiné systémy pro uložení dat.

Klíčovým požadavkem v této fázi je, aby byla všechna data ze zdrojových systémů nahrána v požadovaném čase. S tím se pojí hned několik problémů. Zdrojové systémy mohou být například dočasně nedostupné. Může se také stát, že zdrojový systém není uzpůsoben k tako rozsáhlé extrakci dat a požadovaná zátěž pro něj může být z různých důvodů nepřípustná a je třeba hledat náhradní řešení (zálohy systému, čtení pouze části dat apod.). Dalším problémem, na který je možné při extrakci dat narazit může být například příliš velký objem dat, kdy u těchto procesů není výjimkou objem dat v řádu několika GB denně.

Extrakce probíhá ve dvou fázích [https://www.cs.colostate.edu/etl/papers/Thesis.pdf] . V první fázi probíhá tzv. úplná extrakce. Při této fázi jsou data extrahována poprvé a je tedy nutné je extrahovat kompletně celá. Druhá fáze je tzv. inkrementální. Tato fáze nastává ve chvíli, kdy se ve zdrojových systémech objeví nová nebo modifikovaná data. Nová nebo modifikovaná data je třeba

identifikovat a odlišit od takových, které již procesem extrakce prošla dříve [Kimball]. Identifikace nových dat může být problematická. Můžeme k ní využít tři přístupy.

- Logy v databázi - V této technice mohou být použity logy DBMS. Tyto logy jsou použity pro nalezení přidání nebo změny dat ve zdrojové databázi.
- Triggery - Na každé tabulce ve zdrojové databázi jsou vytvořeny triggery, které jsou automaticky spuštěny při přidání či změně dat ve zdrojové databázi pomocí DML (Data Manipulation Language).
- Časová razítka - Některé databáze používají sloupce pro časová razítka, která specifikují čas ve kterém byl daný řádek naposledy modifikován. Pomocí těchto sloupců lze jednoduše identifikovat změnu ve zdrojovém systému.

Pokud však zdrojovým systémem není relační databáze, není možné takové přístupy použít. Je tedy třeba manuálně nalézt způsob, jak identifkovat přidaná či změněná data a ty následně extrahovat.

Cílem této fáze je tedy identifikovat relevantní informace ve zdrojových systémech a nahrát je do jediné struktury či formátu, která je vhodná pro fázi transformace.

3.2 Transformace dat

Druhým procesem je proces *transformace*. Během transformace jsou data uložena do dočasného datového úložiště (Data Staging Area - DSA), kde jsou očištěna, přeformátována a spojována tak, aby vyhovovala datovému modelu cílového datového skladu. Proces transformace má dvě funkce. První funkcí je čištění dat. Transformační proces identifikuje a opravuje (nebo odstraňuje) existující problémy v datech a připraví je na další proces. Cílem je předejít snaze o transformaci neúplných, či chybných dat, které se mohou v datech vyskytovat. Data by během čištění měla být posouzena jak ze syntaktického, tak sémantického hlediska tak, aby byla validní vzhledem ke zdrojovým podmínkám. Během tohoto kroku je možné aplikovat různé přístupy k posouzení kvality dat, sloužící k detekování problémů, které se mohou v datech vyskytovat. Data jsou kontrolována pomocí různých kvalitativních pravidel, které jsou schopné detekovat poškozená data jak ze syntaktického, tak sémantického hlediska. V tabulce 3.1 je možné vidět různé příklady těchto kontrol s příklady dat, která porušují pravidla kontroly.

Kontrola	Příklad porušení pravidel
validní hodnoty	datum_narozeni=70045 není povolený formát data narození
unikátnost	stejné rodné číslo RC='123456789' přítomno pro dvě osoby
chybějící hodnota	hodnota pohlaví je u některých záznamu 'null'
existující reference	referencovaná nemocnice s identifikátorem '1002' neexistuje
závislost hodnot	mesto=plzen neodpovídá hodnotě psc=55000

Tabulka 3.1: Příklady porušení kvalitativních pravidel

Při procesu čištění je nezbytné definovat výstup těchto kontrol. Poškozená či neúplná data nestačí pouze detekovat, ale je nutné vědět jak s nimi naložit. Data mohou být opravena, zahozena či jiným způsobem zpracována tak, aby nenarušila transformační proces.

V druhém kroku jsou poté data přeformátována a transformována tak, aby vyhovovala požadavkům cílového datového skladu. Na data je aplikována sada transformačních pravidel, která poskytují návrháři datového skladu.

Vincent Rainardi. Building a Data Warehouse with Examples in SQL Server. Apress, 1st edition, 2008.

3.3 Zápis dat

Posledním procesem je *zápis* dat. Tento proces zapíše extrahovaná a transformovaná data z dočasného datového úložiště do cílového datového skladu. Procesy pro zápis dat se mohou značně lišit v závislosti na organizačních požadavcích. V některých datových skladech mohou být existující data přepisována novými daty na denní, týdenní nebo měsíční bázi, zatímco jiné datové sklady mohou uchovávat historii dat přidáváním nových dat v pravidelných intervalech. Komponenta pro zápis dat je často implementována pomocí procesů, které v celku, nebo inkrementálně nahrávají data z dočasného datového úložiště do cílového datového skladu. Během úplného zápisu jsou tak zapsána všechna data transformována v dočasném datovém úložišti. Při inkrementální zápisu jsou inkrementálně zapisována modifikovaná či přidaná data na základě logů, triggeru nebo časových razítek v dočasném datovém skladu.

3.4 ETL nástroje pro zpracování geografických dat

V dnešním světě existuje velké množství různých zdrojů prostorových dat. Data těchto zdrojů se značně liší. Mohou mít odlišné datové struktury, formy zápisů dat a velké množství dalších rozdílů. Iniciativa, která vznikla na území Evropy a která si klade za cíl unifikaci dat evropských států, se nazývá direktiva INSPIRE - Infrastruktura pro prostorové informace v Evropě (Infrastructure for Spatial Information in the European Community).

3.4.1 INSPIRE

INSPIRE je iniciativou Evropské komise. Stejnojmenná směrnice Evropské komise a Rady si klade za cíl vytvořit evropský legislativní rámec potřebný k vybudování evropské infrastruktury prostorových informací. Stanovuje obecná pravidla pro založení evropské infrastruktury prostorových dat zejména k podpoře environmentálních politik a politik, které životní prostředí ovlivňují. Hlavním cílem INSPIRE je poskytnout větší množství kvalitních a standardizovaných prostorových informací pro vytváření a uplatňování politik Společenství na všech úrovních členských států. **https://geoportal.gov.cz/web/gu**
inspire

Direktiva INSPIRE ukládá obecná pravidla pro uložení prostorových dat všech členských států Evropské Unie tak, aby byla vzájemně interoperabilní a použitelná širokou veřejností k různým prostorovým analýzám, plánování a jiných procesů souvisejícími s prostorovými daty. Členské státy dokážou poskytovat prostorová data popisující jejich území, avšak data nejsou unifikovaná a nelze k nim stejným způsobem přistupovat. Pro jednotlivé členské státy je tedy nutné data transformovat tak, aby direktivě INSPIRE vyhovovala. Proces této transformace se nazývá *harmonizace dat*.

3.4.2 Harmonizace dat

Proces harmonizace dat umožňuje kombinovat data z různých heterogenních zdrojů (regionálních datasetů) do integrovaných, konzistentních a jednoznačných souborů dat (Evropských datasetů). Takové datasety mohou být poté jednoduše a hlavně unifikovaně využívány v kombinaci s dalšimi harmonizovanými daty jak k prohlížení, tak k různým analytickým procesům. Proces harmonizace dat je komplexní úlohou, která nemá žádné univerzální řešení, které by pokrylo všechny možné situace. Konkrétní informační systém je

vždy určen mnoha faktory, jako je například uložení dat, jejich velikost a způsob harmonizace.

Nejznámějším přístupem k harmonizaci dat, který data dokáže převádět do cílené formy, je pětikrokový harmonizační proces, jinak nazývaný prostorové ETL. Tento proces má, jak název napovídá, pět kroků. Těmito kroky jsou:

1. porozumění teorie harmonizace prostorových dat - pochopení technik, které dokážou data transformovat mezi různými datovými strukturami při co nejnižší ztrátě dat

2. porozumění zdrojových dat - hluboké porozumění zdrojové struktury dat, jejich zápisu až na úroveň jednotlivých atributů

3. porozumění cílových dat - hluboké porozumění cílové struktury dat, jejich zápisu až na úroveň jednotlivých atributů

4. definice harmonizačních kroků - definování rozdílů mezi zdrojovou a cílovou strukturou dat, definování schémat a procesů popisující způsob konverze mezi zdrojovou a cílovou strukturou při kterých mohou nastat následující situace:

    - jeden cílový element může být sestaven jedním zdrojovým elementem (1:1)
    - jeden cílový element může být sestaven více zdrojovými elementy (1:M)
    - více cílových elementů může být sestaveno více zdojovými elementy (M:N)

5. realizace - implementace výše zmíněných harmonizačních kroků vlastním nebo vybraným nástrojem.

K realizaci se využívají tři typy software. Prvním typem jsou geografické informační systémy, tedy například software ArcGIS. Druhým typem jsou prostorové relační databáze a jejich funkce. Těmi mohou být např PostgreSQL s databází PostGIS. Třetím typem jsou potom ETL nástroje jako Spatial Data Integrator nebo Hale. Realizace je také možná vlastním nástrojem částečně využívajícím již existující nástroje.

Výběr správného typu software pro realizaci harmonizace dat závisí na konkrétní úloze, zdroji dat, cílové datové struktuře a mnoha dalších faktorech.

http://cdn.safe.com/resources/fme/Data-Harmonization-Principles-INSPIRE.pdf https://link.springer.com/article/10.1186/s40965-017-0015-6 https://www.researchgate.net/publication/268464844_TOWARDS_INTER STEPS_HARMONIZATION_FRAMEWORK

Specifikum ETL pro zpracování geografických dat oproti obecnému ETL je práce se specifickými, mnohdy komplikovanějšími, datovými strukturami, kde je při konverzi nutné zachovat topologii dat. Současně je součástí konverze i práce s různými souřadnicovými systémy.

Harmonizace dat je tedy souborem transformačních procesů, který kombinuje data z více zdrojů. Práce k převodu dat ze zdrojové do cílové datové struktury využívá ETL přístup, kde jsou definovány transformační kroky, podle kterých jsou data převedena. Jedná se tak pouze o část harmonizace, jelikož využívá pouze jeden zdroj dat.

4 Datové struktury zdrojového a cílového nástroje

Tato kapitola se věnuje datovým strukturám zdrojového a cílového nástroje. Popisuje tak druhý a třetí krok pětikrokového procesu harmonizace dat popsaného v kapitole ETL. V první části je popsána struktura cílových dat nástroje Traffic Modeller, pro který jsou data připravována. Druhá část věnována zdrojové datové struktuře databáze OpenStreetMap, která může být pro nástroj Traffic Modeller použita.

4.1 Traffic Modeller

Cílem práce je transformovat data silniční sítě tak, aby vyhovovala nástroji Traffic Modeller. Nástroj Traffic Modeller, neboli *TraMod*, je nástroj sloužící k modelování dopravy vyvíjený na Západočeské univerzitě ve spolupráci s organizacemi Plan4All, EDIP, HSRS a RoadTwin. Jejím uživatelům umožňuje simulovat a tak testovat různé dopravní scénáře. Nástroj je službou běžící na serveru a je možné ho užívat bez nutnosti jakékoliv instalace či znalosti jiného software modelujícího dopravu. Nástroj je možné využívat pomocí webové aplikace, která je pro náhled dostupná na `https://intenzitadopravy.plzen.eu/`. Pro aktivní práci v současné době není dostupná široké veřejnosti. Nástroj poskytuje API, což umožňuje vývojářům integrovat funkce nástroje do dalších mobilních či desktopových aplikací. **https://trafficmodeller.com**

Traffic Modeller využívá dopravní model k simulování toku dopravy. Dopravním modelem rozumíme matematický model skutečné dopravy, který určuje hustotu dopravy (počet vozidel) v úsecích dopravní sítě. Tento model je reprezentovaný orientovaným grafem. Ke správnému fungování modelu jsou zapotřebí tři vstupy. Prvním vstupem je dobře definovaná směrovatelná silniční síť, která popisuje přípustné pohyby v síti uvnitř pozorovaného modelu. Tento model musí odpovídat skutečnosti a jakákoliv nesouvislost s reálnou situací zvyšuje odchylku od reálného modelu dopravy. Odchylku simulovaného modelu od reálného určuje GEH statistika **https://content.tfl.gov.uk/traffic-modelling-guidelines.pdf**. Krom to-

pologie je také důležité, aby parametry dopravního modelu odpovídaly skutečnosti. Nepřesnosti mohou mít za následky nerealistické simulace. Tou může být například hromadící se doprava, kvůli chybnému určení maximální rychlosti na určitém úseku vozovky v simulovaném modelu. Druhým vstupem jsou pak generátory dopravy, které reprezentují přírůstky dopravy v různých místech modelu. Třetím vstupem jsou pak kalibrační data modelu, které však nejsou předmětem diplomové práce a nejsou zde tedy popsána **https://iccgis2020.cartography-gis.com/8ICCGIS-Vol1/8ICCGIS_Proceedings_Vol**

4.1.1 Datová struktura

Datová struktura nástroje je uložena v databázi popsané v dokumentaci datové struktury nástroje Traffic Modeller na serveru Gitlab *https://gitlab.com/tramod/tramod-data-model*. Databázový model je rozdělen na několik částí. První částí je model dopravní infrastruktury popsaný pomocí tabulek uzlů a hran. Různá omezení pohybu v této síti jsou zanesena v tabulce zakázaných směrů. Další částí jsou pak socioekonomická data popsána generátory dopravy a kalibračními daty.

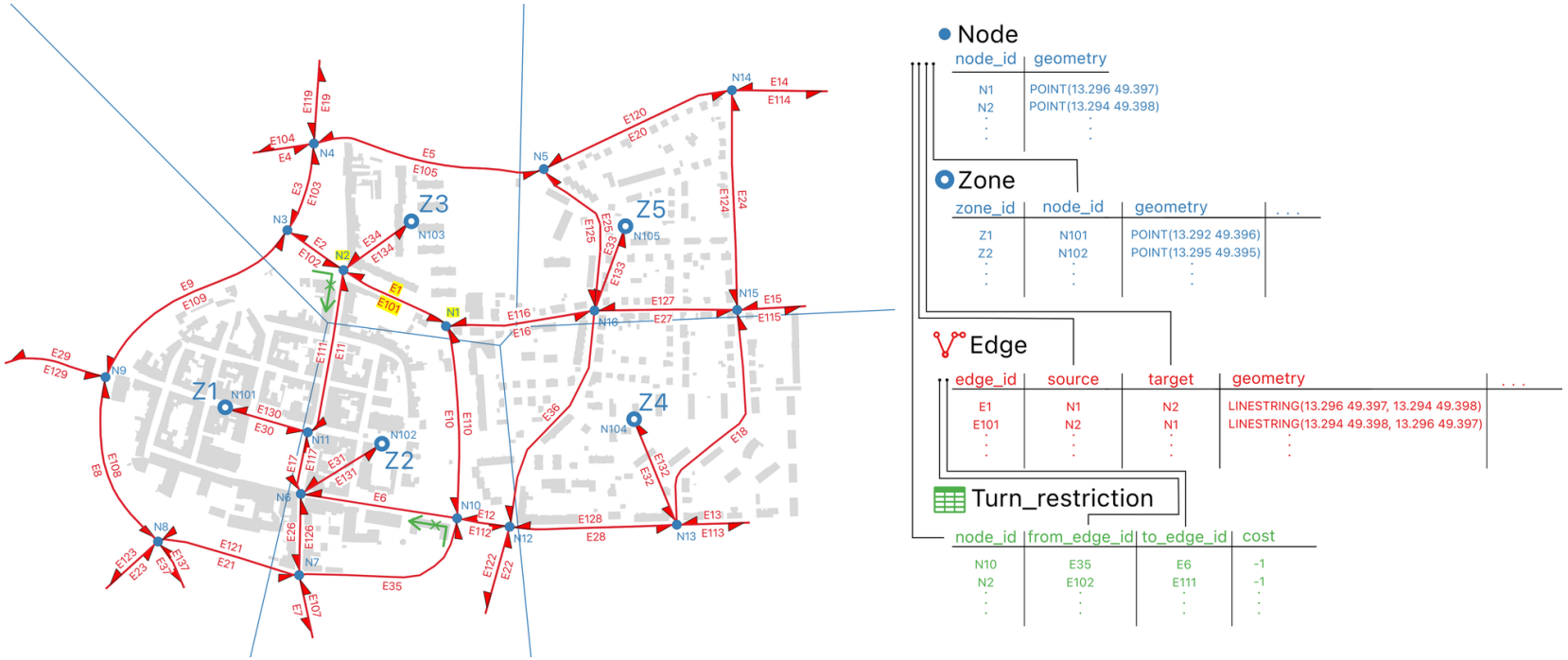
- *node* - tabulka uzlů
- *edge* - tabulka hran
- *zone* - zóny reprezentující generátory dopravy
- *turn_restriction* - tabulka zákazů odbočení

Reprezentace dat pomocí uvedených tabulek je znázorněna na obrázku 4.1 převzatého z **Potenciál interaktivního dopravního modelování pro územní plánování měst, Diplomová práce, Bc. Petr Trnka, 2022, vedoucí práce: Karel Jedlička KGM**.

Nástroj navíc obsahuje tabulku *odm* (origin-destination matrix). Tato tabulka slouží ke kalibraci dat. Kalibrace dat není předmětem této práce a proto tato tabulka není dále popisována.

Uzly

Tabulka uzlů v nástroji označuje křižovatky. Na každém místě, kde se komunikace fyzicky kříží musí existovat v datech nástroje Traffic Modeller právě jeden uzel. Uzel obsahuje svůj identifikátor a také hodnotu geometrie, která v sobě nese geografické údaje o jeho poloze. Vrchol také existuje v bodě konce vozovky (např. na konci slepé ulice). Zde jsou uvedeny jednotlivé sloupce tabulky uzlů:



Obrázek 4.1: Grafické schéma TraMod

- *node_id* - identifikátor uzlu
- *geometrie* - geometrie typu *Point* v souřadnicovém systému WGS84

Hrany

Tabulka hran (*edge*) obsahuje všechna spojení mezi uzly. Každá komunikace od křižovatky ke křižovatce je zaznamenána právě jednou hranou. Každá hrana obsahuje svůj identifikátor, referenci na zdrojový uzel, cílový uzel a geometrii. Tvoří tedy jednosměrnou orientovanou hranu. Pokud mezi dvěma uzly A a B existuje obousměrná komunikace, musí v tabulce existovat dvě hrany s unikátními identifikátory. První záznam bude mít uvedený uzel A jako zdrojový a uzel B jako cílový a druhý záznam bude mít uvedený uzel B jako zdrojový a uzel A jako cílový. Pokud je silnice mezi uzly $A.B$ jednosměrná, existuje pouze jedna hrana, která obsahuje uzel A jako zdrojový a B jako cílový. Hodnota geometrie určuje polohu a tvar křivky, která příslušnou komunikaci reprezentuje. Pokud vozovka obsahuje více pruhů, je stále reprezentována pouze jednou hranou. Jedna hrana v databázi je tedy svým identifikátorem, identifikátorem zdrojového a cílového uzlu. Obsahuje také několik dalších záznamů, které jsou uvedeny v následujícím výčtu:

- *edge_id* - identifikátor uzlu
- *source* - geometrie elementu
- *target* - geometrie elementu
- *capacity* - počet vozů, které mohou hranou projet za hodinu
- *cost* - cena za projetí této cesty (čas)
- *isvalid* - atribut sloužící pro plánované cesty (true = cesta existuje, false = cesta je plánována)
- *turn_restriction* - zákazy, či příkazy odbočení v textové podobě
- *speed* - maximální povolená rychlost v km/h
- *road_type* - třída vozovky
- *geometry* - geometrie typu *LineString* v souřadnicovém systému WGS84

Zakázané směry

Tabulka zakázaných směrů (turn_restriction) obsahuje záznamy pro všechny zákazy odbočení na mapě. Obsahuje identifikátor křižovatky (uzlu), ke kterému náleží, identifikátor zdrojové hrany a identifikátor cílové hrany. Pokud tedy nelze z komunikace (hrany) A na křižovatce (uzlu) X odbočit na komunikaci B, bude v tabulce uveden záznam s identifikátorem uzlu X, zdrojovou hranou A a cílovou hranou B. Pokud je na na jedné křižovatce zákazů více, bude pro každý zákaz v tabulce jeden záznam. Nástroj Traffic Modeller neobsahuje způsob pro uchování příkazů odbočení. Pokud se na nějaké křižovatce vyskytuje přikázané odbočení, budou zakázány všechny ostatní možnosti odbočení a tyto zákazy budou zaneseny do tabulky. Zde jsou uvedeny jednotlivé sloupce tabulky zakázaných směrů.

- *node_id* - identifikátor uzlu
- *from_edge_id* - zdrojová hrana
- *to_edge_id* - cílová hrana
- *cost* - cena za projetí křižovatky v tomto směru

Generátory dopravy

Tabulka zón *zone* reprezentuje tzv. *generátory dopravy*. Generátor dopravy je odhad počtu vozidel, které budou denně vyjíždět z oblasti jím vymezeném. Počet vozidel je odhadován na základě počtu, plochy a typu budov, které leží v jedné vymezené zóně. Generátor jednotlivých oblastí je poté přiřazen vhodné křižovatce, kterou poté nástroj TraMod využívá k modelování dopravních situací. Jednotlivé sloupce tabulky zón jsou popsány

- *zone_id* - identifikátor generátoru dopravy
- *node_id* - identifikátor křižovatky s přiřazeným generátorem
- *trips* - odhadovaná hodnota generátoru dopravy
- *geometry* - geometrie centroidu zóny typu *Point* v souřadnicovém systému WGS84

Jak již bylo řečeno tabulky databáze napovídají, silniční síť je v nástroji Traffic Modeller reprezentována grafovou strukturou. Vrcholy tohoto

grafu označují křižovatky a hrany grafu reprezentují silnice, které křižovatky spojují. Taková reprezentace umožňuje nástroji modelovat dopravu pomocí algoritmů užívaných pro různé grafové problémy, jako je například hledání nejkratší cesty mezi dvěma uzly.

4.2 OpenStreetMap

Data pro nástroj Traffic Modeller mohou být vytvořena manuálně pro každé území zájmu, avšak je také možné jej naplnit veřejně dostupnými daty. OpenStreetMap (*OSM*) je veřejná databáze poskytující zeměpisná data. Projekt v roce 2004 založil Steve Coast. Je tvořena komunitou uživatelů, kteří přidávají a udržují data o silnicích, cestách, kavárnách, železničních stanicích a mnohém dalším po celém světě. Mezi přispěvateli lze najít profesionály z oblasti GIS, techniky spravující OSM, amatérské kreslíře map, či humanitární pracovníky. OSM tvoří otevřená data, která mohou být využívána k libovolným účelům pod licencí ODbL. Open Database License (ODbL) je copyleftová ("Share Alike") licence, která umožňuje uživatelům svobodně sdílet, upravovat a používat databázi za podmínky poskytnutí stejné svobody ostatním uživatelům. Data tak mohou být použita, pokud je uvedeno autorství OSM a jeho přispěvatelů. Pokud jsou data jakýmkoliv způsobem upravována, či rozšiřována, je možné výsledek šířit pod stejnou licencí **https://opendatacommons.org/licenses/odbl/1-0/**.

4.2.1 Export dat

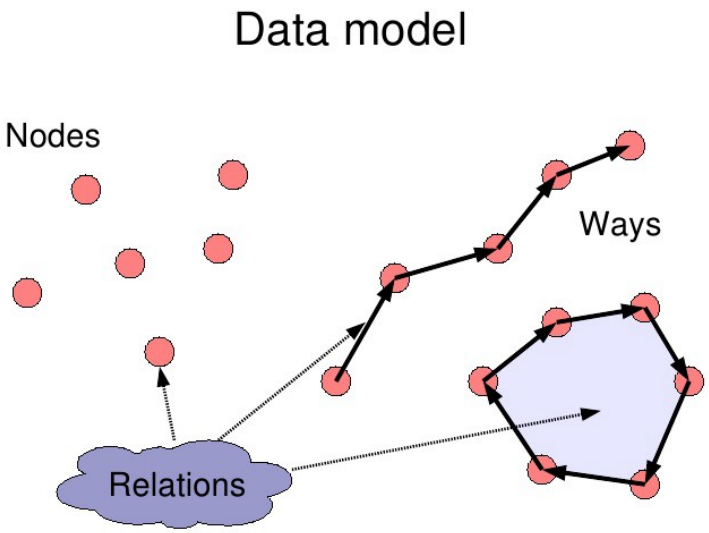
Data OSM lze získat různými způsoby. Samotný portál `openstreetmap.org` umožňuje export dat do souboru formát PBF nebo OSM. Formát PBF je komprimovaným formátem, zatímco OSM je založen na formátu XML. Portál samotný omezuje export do souboru na oblast maximálně 0.25°čtverečních, nebo na oblast obsahující nejvýše 50 000 bodů. Existují ale i jiné portály, které lze pro získání dat využít.

Jedním z nich je portál `geofabrik.de`, který spravuje pravidelně aktualizované extrakty kontinentů zemí a vybraných měst. Omezením portálu je možnost stažení pouze již vymezených extraktů a není tedy možné ručně vybrat oblast. Dalším nástrojem pro export dat je portál bbbike `extract.bbbike.org`, který umožňuje vymezit požadované území. Oproti portálům OSM a GeoFabrik navíc umožňuje vybrat data uvnitř vymezeného polygonu a není tak omezený na obdelníkový tvar území. Jeho omezením je maximální oblast o velikost 25 000 000 kilometrů čtverečních nebo maximální velikost požadovaného souboru 512MB.

https://alga.win.tue.nl/tutorials/openstreetmap/

4.2.2 Datová struktura

Data v OSM jsou uložena v jednoduché datové struktuře, která je tvořena třemi typy elementů. Těmito typy jsou *nodes* (uzly), *ways* (cesty) a *relations* (relace). Všechny tři typy elementů mohou mít přiřazen jeden nebo více atributů (*tags*), které popisují význam jednotlivých elementů a tvoří tak popisnou složku dat. Dle části 2.2 se tak jedná o hierarchický model. Pro každý typ elementu je vybrán příklad jeho zápisu ze souboru ve formátu OSM.



Obrázek 4.2: Datová struktura OSM

Uzly (*nodes*) reprezentují jednobodové elementy v mapě. Jsou definovány svým identifikátorem a zeměpisnou šířkou a délkou. Mohou reprezentovat elementy na mapě, které jsou samostatně stojící a k jejich popisu uzel stačí. Mohou tak být reprezentovány například lavičky v parku, lampy apod. Význam takových uzlů je určen právě pomocí značky. Uzly jsou také využívány k určení tvaru vícebodových elementů. V takovém případě nemusí všechny uzly obsahovat značku, která by určila jejich význam.

Příklad záznamu uzlu v souboru formátu OSM.

```
<node id="32665891" lat="49.7534179" lon="13.5889677"/>
```

Cesty (*ways*) reprezentují nejednobodové elementy a jsou tvořeny sledem (seřazeným listem) referencovaných bodů. Sled může kvůli omezení OSM ob-

sahovat dva až dva tisíce uzlů. Uzly mají tedy dvojí funkci. Jsou buď samostatně stojícími elementy, či součástí cest. Cesty mohou reprezentovat dva typy elementů. Prvním typem elementu jsou neuzavřené cesty (křivky). Tedy cesty jejichž první a poslední uzel se neshodují. Je možné takto reprezentovat silnice, různé lesní cesty, řeky, či vedení elektrického napětí. Druhým typem elementu, který je možné cestou reprezentovat jsou uzavřené cesty (hranice polygonu). V takovém případě je první a poslední uzel v cestě stejný. Tímto způsobem lze do OSM zanést například budovy, vodní plochy, či lesy. Význam cest je opět uchováván pomocí značek. Důležité je mít na paměti, že i některé silnice mohou být zaneseny pomocí cesty, která bude mít první a poslední uzel stejný. Typickým příkladem je kruhový objezd. K rozlišení je nutné prozkoumat značky, které v takové chvíli musí jasně určit, zda se jedná o nějaký element vyjadřující plochu, či nějaký cyklický element.

Příklad záznamu cesty v souboru formátu OSM.

```
<way id="225524966">
    <nd ref="2342966753"/>
    <nd ref="2342966733"/>
    <nd ref="2342966748"/>
    <nd ref="2342966753"/>
</way>
```

Třetím typem elementu jsou relace (*Relations*). Relace jsou víceúčelovou datovou strukturou, která poisuje vztah mezi dvěma či více elementy (uzly, cestami, či jinými relacemi). Typickým příkladem může být seznam silnic, které dohromady tvoří dálnici, či trasu linky městské hromadné dopravy. Dalším příkladem může být zákaz odbočení, který vyjadřuje ze které cesty není možné na kterou cestu odbočit. Jiným přikladem může být vícebodový polygon, který reprezentuje vnější hranici nějakého plošného elementu a nevyplněná plocha v něm je popsána druhým polygonem tvořícím vnitřní hranici. Význam těchto relací je vždy určen pomocí značek. Typicky bude mít relace značku *type*, která určí typ této relace. Relace jsou vyjádřeny seřazeným listem uzlů, cest, či jiných relací.

Příklad záznamu relace v souboru formátu OSM.

```
<relation id="13825735">
    <member type="way" ref="1033228210" role="inner"/>
    <member type="way" ref="244532456" role="outer"/>
</relation>
```

K určení elementů, které cesty, relace a jednotlivé body reprezentují se využívají tzv. atributy (tags). Značky je možné přiřadit k jednotlivým uzlům, celým cestám, či relacím. Atributy obsahují klíč a hodnotu. Klíč je užíván

k definování názvu objektu a hodnota k definování jeho hodnoty. Některé atributy hodnotu nepotřebují, užívá se potom klíčové slovo *yes*.

Příklady atributů elementů:

- jednobodové - *shop=supermarket*
- neuzavřené cesty (křivky) - *highway=motorway*
- uzavřené cesty (polygony) - *building=yes*

Příklad záznamu atributů uvnitř relace v souboru formátu OSM:

```
<relation id="13525552">
    <member type="way" ref="991947215" role="outer"/>
    <member type="way" ref="1009915444" role="outer"/>
    <tag k="building" v="yes"/>
    <tag k="public_transport" v="station"/>
    <tag k="type" v="multipolygon"/>
</relation>
```

Portál openstreetmap.org umožňuje zobrazit značky jednotlivých elementů. Značky jednotlivých elementů jsou uživateli zobrazeny po kliknutí na element pomocí funkce *Průzkum prvků*.

5 Návrh nástroje OSM2TraMod

V předchozí kapitole jsou popsány zdrojové a cílové datové struktury. V této kapitole je popsán návrh konverzního nástroje, který dokáže převést zdrojová data ve formátu OSM na cílový formát nástroje Traffic Modeller. Jsou tak postupně popsány kroky harmonizace, tedy třetí třetí krok pětikrokového procesu harmonizace dat. První část kapitoly je věnována přípravě dat s využitím externích nástrojů a druhá část kapitoly poté návrhu vlastního konverzního nástroje, který převede připravená data na datovou strukturu cílového nástroje.

5.1 Využití externích modulů

K naplnění nástroje Traffic Modeller je potřeba z dat OSM získat data o silniční síti a následně data, ze kterých bude možné vypočítat generátory dopravy. Tato data je možné získat pomocí externích nástrojů, které jsou v této části popsány.

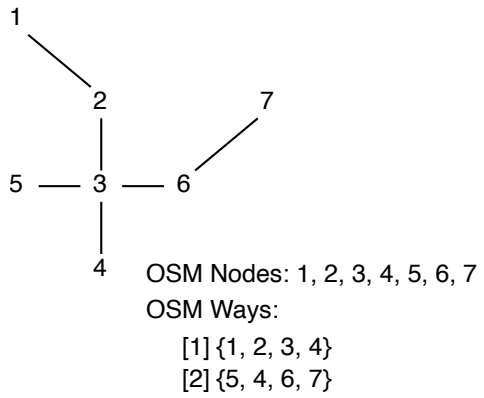
5.1.1 Silniční síť

K využití dat OSM v nástroji Traffic Modeller je důležité prozkoumat způsob, jakým jsou v OSM reprezentována data týkající se silniční sítě. Data v OSM jsou reprezentována následujícím způsobem. Každá pozemní komunikace je reprezentována jednou nebo více cestami (mohou být součástí většího spoje). Pokud se dvě komunikace fyzicky kříží tak, že je možné dostat se z jedné komunikace na druhou, existuje v místě jejich překřížení uzel. Pokud se však cesty fyzicky nekříží, například z důvodu existence mostu, který je druhou cestou podjížděn, v bodě překřížení těchto hran uzel neexistuje. Pokud má pozemní komunikace více pruhů, které nejsou odděleny žádnou fyzickou bariérou, je v datech zanesena jako jedna cesta. Pokud mezi sebou jednotlivé pruhy mají fyzickou bariéru (dálnice), jsou v OSM zaneseny jako dvě jednotlivé cesty.

Dále je ke správnému fungování důležité určit, jakým směrem cesty vedou. Cesty jsou reprezentovány sledem dvěma a více bodů. Pokud existuje komunikace mezi uzly A a B, bude existovat cesta $\{A,B\}$ nebo cesta $\{B,A\}$,

ale nikoliv obě. Takovou cestu můžeme v grafu poté chápat jako neorientovanou hranu mezi dvěma uzly. Pokud je však komunikace jednosměrná, cesta bude označena značkou *oneway=yes*. Takové označení využívají veškeré jednosměrné komunikace, včetně obousměrných komunikací oddělených fyzickou bariérou. Pokud cesta obsahuje tuto značku, její směr je poté dán pořadím uzlů, které cestu tvoří.

Základní vlastností pro využití dat o pozemních komunikacích nástrojem Traffic Modeller je jejich směrovatelnost (z angl. *routability*). Směrovatelnost tak znamená, že je možné se z libovolného uzlu dostat pomocí cest do jiného libovolného uzlu. Data OSM však tento předpoklad nesplňují. Nedostatek je znázorněn na obrázku 5.1.



Obrázek 5.1: Nesměrovatelná síť

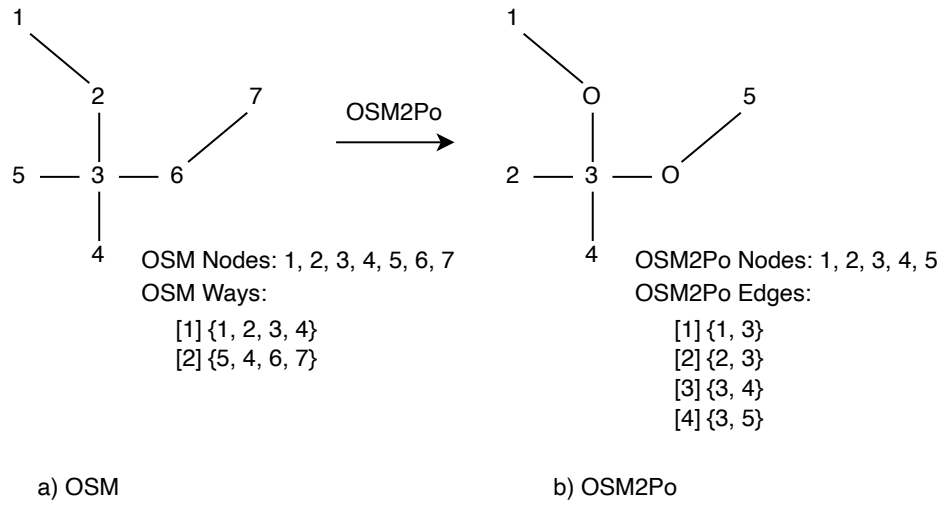
Na obrázku 5.1 je vidět 7 bodů: *{1, 2, 3, 4, 5, 6, 7}*. Tyto body reprezentují body na silnici. V datech jsou také zaneseny dvě cesty. Cesta *1: {1, 2, 3, 4}* a cesta *2: {5, 3, 6, 7}*. Takováto data tedy mohou reprezentovat jednoduchou křižovatku. Díky první cestě například víme, jakým způsobem se dostat z bodu *{1}* do bodů *{2, 3}* nebo *{4}*. Problém vzniká ve chvíli, kdy bychom se chtěli z bodu *{1}* dostat například do bodu *{7}*. Je zřejmé, že do bodu *{7}* je možńe se dostat přes bod *{3}* a cesta by tedy byla složená z bodů *{1, 2, 3, 6, 7}*. Bohužel však v datech Open Street Map taková cesta být nemusí a v datech tedy neexistuje způsob, jakým se z bodu *{1}* dostat do bodu *{7}*. Taková síť je tedy nesměrovatelná.

Základním předpokladem nístroje Traffic Modeller je však směrovatelná síť. Tedy síť, ve které bude možné najít cestu z libovolného zdrojového bodu, do libovolného cílového bodu. Prvním problémem je tedy převod nesměrovatelné silniční sítě nástroje OSM na směrovatelnou silniční síť.

Směrovatelná silniční síť - OSM2Po

Problém směrovatelnosti silniční sítě je možné řešit dvěma způsoby. Prvním způsobem je najít volně dostupný nástroj, který směrovatelnou síť z dat OSM vytvoří a s jeho výstupem následně pracovat. Druhým a mnohem pracnějším způsobem je pak vytvořit směrovatelnou síť manuálně.

Nástrojem, který z nesměrovatelné silniční sítě dat OSM dokáže vytvořit směrovatelnou síť je nástroj *OSM2Po*. Vytvořil jej německý vývojář Carsten Möller a je volně dostupným nástrojem. Tento nástroj je současně konvertorem a směrovacím enginem. Dokáže parsovat data z OSM a vytvořit z nich směrovatelnou síť. Proces vytvoření směrovatelné sítě z předchozího příkladu je vidět na obrázku 5.2:



Obrázek 5.2: OSM2Po proces

OSM2Po nejdříve nalezne uzly, kde se komunikace fyzicky kříží (rozdělují, slučují, končí). Tyto uzly zaznamená a poté vytvoří cesty hrany mezi všemi sousedními uzly. Původní počet uzlů dat OSM byl tedy redukován ze 7 na 5. Vrcholy, které jsou vynechány v OSM slouží pouze k určení tvaru cesty. Nástroj OSM2po tento tvar ukládá do vlastního záznamu s geometrií cesty mezi body. Vznikly tedy 4 cesty a je nyní jasné, že se s využitím jedné cesty nebo jejich kombinací je možné dostat z libovolného bodu do libovolného jiného bodu. Například chceme-li se jako v předchozím případě dostat z bodu *{1}* do bodu *{7}* (nyní bodu *{5}*), využijeme kombinaci cest *{1, 3}{3, 5}*. Takováto síť je tedy směrovatelná a je možné ji použít pro nástroj

Traffic Modeller.

Nástroj po svém spuštění vytvoří jednoduchý web server, ze kterého ,po přidání parametrů typu *GET* do URL, dokáže obsloužit nejběžnější případy použití směrovatelných enginů. Dokáže nalézt nejbližší uzel jinému uzlu, nejkratší cestu mezi dvěma uzly či dokáže řešit problém obchodního cestujícího po zadání požadovaných uzlů. Nástroj take umožňuje exportovat data se silniční sítí do dvou SQL souborů, pomocí kterých je možné nahrát data do *PostGIS* databáze. Těmito soubory jsou:

- uzly - v souboru *<název_území>_2po_vertex.sql*
- hrany - v souboru *<název_území>_2po_4pgr.sql*

Každý z těchto souborů tvoří příslušnou tabulku v databázi.
Vrcholy:

- *id* - identifikátor uzlu
- *clazz* - typ uzlu
- *osm_id* - původní id elementu z osm
- *osm_name* - původní název elementu v osm
- *ref_count* - počet referencí tohoto uzlu
- *restrictions* - textová reprezentace zákazů odbočení přes tento uzel
- *Geometry* - Point geometrie elementu v souřadnicovém systému WGS84

Záznamy v tabulce uzlů reprezentují veškerá místa, kde se vozovka fyzicky kříží či končí. Reprezentují tedy veškeré křižovatky, místa kde se vozovky rozdělují, slučují, sjezdy z dálnic, konce slepých ulic a další. Neukládá však jiné uzly. Narozdíl od OSM není tvar cest reprezentován uzly, ale každá hrana (vozovka spojující dva křižovatky) obsahuje vlastní záznam s geometrií, ve které je tento tvar uložen. Dále nejsou exportovány uzly netýkající se silnic. Ve výstupním souboru se tedy vyskytuje méně záznamů uzlů, než ve zdrojovém souboru ve formátu OSM.

Důležitou vlastností, kterou je možné najít v každém záznamu o uzlu, je záznam o tzv. *Restrictions*, nebo-li zákazech. Tento záznam popisuje zákazy či příkazy odbočení, které se na této křižovatce vyskytují. Díky tomuto záznamu bude možné jednoduše naplnit tabulku *Turn_Restrictions* v cílovém nástroji Traffic Modeller.

Hrany:

- *id* - identifikátor hrany
- *osm_id* - původní id elementu v OSM
- *osm_name* - původní název elementu v OSM
- *osm_meta* - původní meta informace z OSM
- *osm_source_id* - původní id počátečního uzlu v OSM
- *osm_target_id* - původní id koncového uzlu v OSM
- *clazz* - číselný kód typu vozovky
- *flags* - typ dopravy
- *source* - id zdrojového uzlu
- *target* - id koncového uzlu
- *km* - délka úseku v km
- *kmh* - maximální povolená rychlost v km/h
- *cost* - cena za přejetí tohoto úseku (km/kmh)
- *reverse_cost* - cena za přejetí úseku v opačném směru
- *x1* - x souřadnice počátečního uzlu v souřadnicovém systému WGS84
- *y1* - y souřadnice počátečního uzlu v souřadnicovém systému WGS84
- *x2* - x souřadnice koncového uzlu v souřadnicovém systému WGS84
- *y2* - y souřadnice koncového uzlu v souřadnicovém systému WGS84
- *Geometry* - LineString geometrie v souřadnicovém systému WGS84

Záznamy v tabulce hran reprezentují veškeré úseky na vozovce od jednoho uzlu k druhému. Každý záznam o úseku z vozovky je definován svým identifikátorem, zdrojovým bodem a cílovým bodem. V seznamu jsou uvedeny další informace, které v sobě záznam uchovává. Důležitou informací je směr daného úseku. Pokud je vozovka jednosměrná, vede od zdrojového k cílovému uzlu, hodnota *cost* určuje čas potřebný k přejetí daného úseku a hodnota *reverse_cost* je nastavena na *1000000.0*. Pokud je vozovka obousměrná, má nastavené obě hodnoty *cost* i *reverse_cost*, kde hodnota *cost*

Hrany		
TraMod	**OSM2Po ekvivalent**	**Pozn.**
edge_id	edges.id	...
source	edges.source	...
target	edges.target	...
capacity	...	...
cost	edges.cost / reverse_cost	...
isvalid	...	default = true
turn_restriction	vertexes.turn_restrictions	
speed	edges.kmh	...
road_type	edges.clazz	...
geometry	edges.geometry	...

Tabulka 5.1: Srovnání tabulek hran

Vrcholy		
TraMod	**OSM2Po ekvivalent**	**Pozn.**
node_id	vertexes.id	...
geometry	geometry	...

Tabulka 5.2: Srovnání tabulek vrcholů

značí potřebný čas k přejetí úseku od zdrojového k cílovému bodu a hodnota *reverse_cost* značí čas potřebný k přejetí vozovky od cílového bodu ke zdrojovému. Tvar křivky reprezentující vozovku není narozdíl od OSM reprezentován několika body, avšak samostatnou hodnotou geometrie.

Při použití *OSM2Po* je tedy splněn předpoklad, že z jakéhokoliv vrcholu je možné se dostat pomocí cest, či jejich kombinace do libovolného jiného vrcholu v síti. Je tedy vyřešen problém směrovatelnosti sítě a není potřeba vymýšlet a implementovat vlastní proces, který by z dat nástroje OSM vytvořil směrovatelnou síť.

Srovnání datových struktur

Při pohledu na datovou strukturu nástroje Traffic Modeller a výstupní datovou strukturu nástroje OSM2Po je vidět jasná podobnost. V následujících tabulkách jsou uvedeny jednotlivé sloupce z tabulek nástroje Traffic Modeller a jejich ekvivalent na straně výstupu nástroje OSM2Po.

Tabulky těchto nástrojů si téměř odpovídají, avšak ke správné kompati-

bilitě dat je potřeba tato data ještě dále transformovat. Konkrétní rozdíly a jednotlivé transformace jsou popsány v části 5.2.3. I přes tyto rozdíly se však dá výstupní struktura nástroje OSM2Po považovat za vhodně připravená data pro nástroj Traffic Modeller a tato výstupní data tedy budou použit jako zdroj implementovaného konverzního nástroje.

5.1.2 Generátory dopravy

Generátory dopravy jsou, stejně jako silniční síť, další nezbytnou součástí nástroje Traffic Modeller. Generátory dopravy rozumíme elementy reprezentující přírůstek dopravy v síti. Hodnota tohoto elementu určuje přírůstek dopravy vznikající v místě tohoto elementu. Jelikož nemají generátory žádnou explicitní prostorovou složku, jsou přiřazeny nějakému uzlu (křižovatce) uvnitř oblasti, ze které jsou vypočítány. Hodnota generátoru, tedy hustota generovaného provozu, je počítána na základě různých demografických zdrojů a měla by odrážet lokální parametry, jako je například počet obyvatel, počet škol a další.

Pro účely diplomové práce bylo rozhodnuto, že těmito lokálními parametry bude zastavěná plocha budov v oblasti generátoru. Generátor se tedy bude počítat jako obsah polygonů, které budovy tvoří. Takto vypočítaná hodnota samozřejmě nemůže odrážet reálnou hustotu dopravy, vznikající v místě generátoru. Výpočet se totiž bude lišit pro jiné typy budov, počet jejich pater a dále. Například je jasné, že v místě obytného domu se sedmi patry bude vznikat větší hustota dopravy, než například v místě s pěti rodinnými domy.

Filtrace

Data nástroje OSM reprezentují budovy jako cesty. Tyto cesty využívají atribut elementu *<tag k="building"v="yes"/>*. Ze zdrojového souboru je tedy potřeba získat pouze cesty s tímto atributem a příslušné uzly. Tímto způsobem je možné zásadně redukovat množství dat, které budou použity pro výpočet. Tato redukce bude mít za následek zrychlení běhu výpočetního algoritmu a snížení jeho spotřebované paměti.

K filtraci je možné využít nástroj *osmfilter* dostupný na wiki nástroje OSM na odkazu `wiki.openstreetmap.org/wiki/Osmfilter`. Nástroj *omsfilter* dokáže na základě zadaných parametrů filtrovat nejrůznější data OSM. Tyto parametry jsou k nalezení na uvedeném odkazu. Při spuštění je mu pomocí vstupního parametru předán soubor ve formátu *osm*, následně seznam parametrů pro filtraci a název výstupního souboru. Nástroj načte vstupní

soubor a na základě zadaných parametrů vyfiltruje data a následně vygeneruje zadaný výstupní soubor ve formátu *osm*.

Konverze

Soubory s daty OSM mohou být získány ve dvou formátech. Jedním z formátů je formát *osm*. Tento formát obsahuje textovou reprezentaci dat v XML struktuře. Druhým formátem je formát *pbf*. Ten je pouze komprimovaným formátem původního formátu osm. Není tedy lidsky čitelný. Některé nástroje umí pracovat s oběma formáty, zatímco některé (např *osmfilter*) umí pracovat pouze s jedním z nich. Je tedy vhodné mít možnost převodu mezi těmito formáty. K převodu formátů je možné použít volně dostupný nástroj *osmium*. Nástroj je dostupný přes portál GitHub na adrese `github.com/osmcode/osmium-tool`. Nástroj dokáže libovolně převádět mezi těmito formáty.

5.2 Návrh vlastního nástroje

V této části je popsán návrh vlastního konverzního nástroje. Tento nástroj využije data popsaná v části přípravy dat jako vstup. Výstupem jsou soubory použitelné pro načtení dat do databáze nástroje Traffic Modeller. Je zde popsán celkový proces konverze a jednotlivé algoritmy, které jsou v nástroji implementovány.

5.2.1 Proces konverze

Celkový proces transformace přímo vychází z části o přípravě dat. Po stažení dat z OSM je možné pomocí zmíněných nástrojů data připravit, což zjednodušší finální transformace implementované ve vlastním nástroji. První částí je připravit data silniční sítě. K tomu je použit nástroj OSM2Po. Výstupem nástroje OSM2Po jsou *SQL dump* soubory reprezentující směrovatelnou silniční síť. *SQL dump* je textový soubor obsahující popis struktury i obsahu tabulek ve formě série SQL příkazů. Z těchto souborů je následně možné načíst data uzlů a hran, které společně tvoří silniční síť.

Druhou částí je poté připravit data pro výpočet generátorů dopravy. Jak bylo zmíněno v kapitole o přípravě dat, výpočet generátorů dopravy využívá data o budovách. Data ze staženého OSM jsou tedy redukována filtrací. K filtraci je využit nástroj *osmfilter*. Tento nástroj dokáže zpracovat data pouze ve formátu *osm*. Pokud jsou tedy zdrojová data stažena ve formátu *pbf*, je potřeba soubor konvertovat nástrojem *osm*.

Takto připravená data budou sloužit jako vstup implementovaného nástroje, jehož výstupem budou *SQL Dump* soubory pro načtení dat do nástroje Traffic Modeller. Celkový proces je znázorněn na obrázku 5.3.

Proces tedy zahrnuje následující kroky:

- stažení zdrojového souboru z portálu OSM ve formátu *osm* nebo *pbf*
- vytvoření směrovatelné sítě pomocí nástroje *OSM2Po*
- případná konverze formátu dat nástrojem *osmium*
- filtrace dat nástrojem *osmfilter*
- proces zbylých transformací nástrojem vytvořeným diplomové práce
- import dat do databáze nástroje Traffic Modeller

5.2.2 Docker

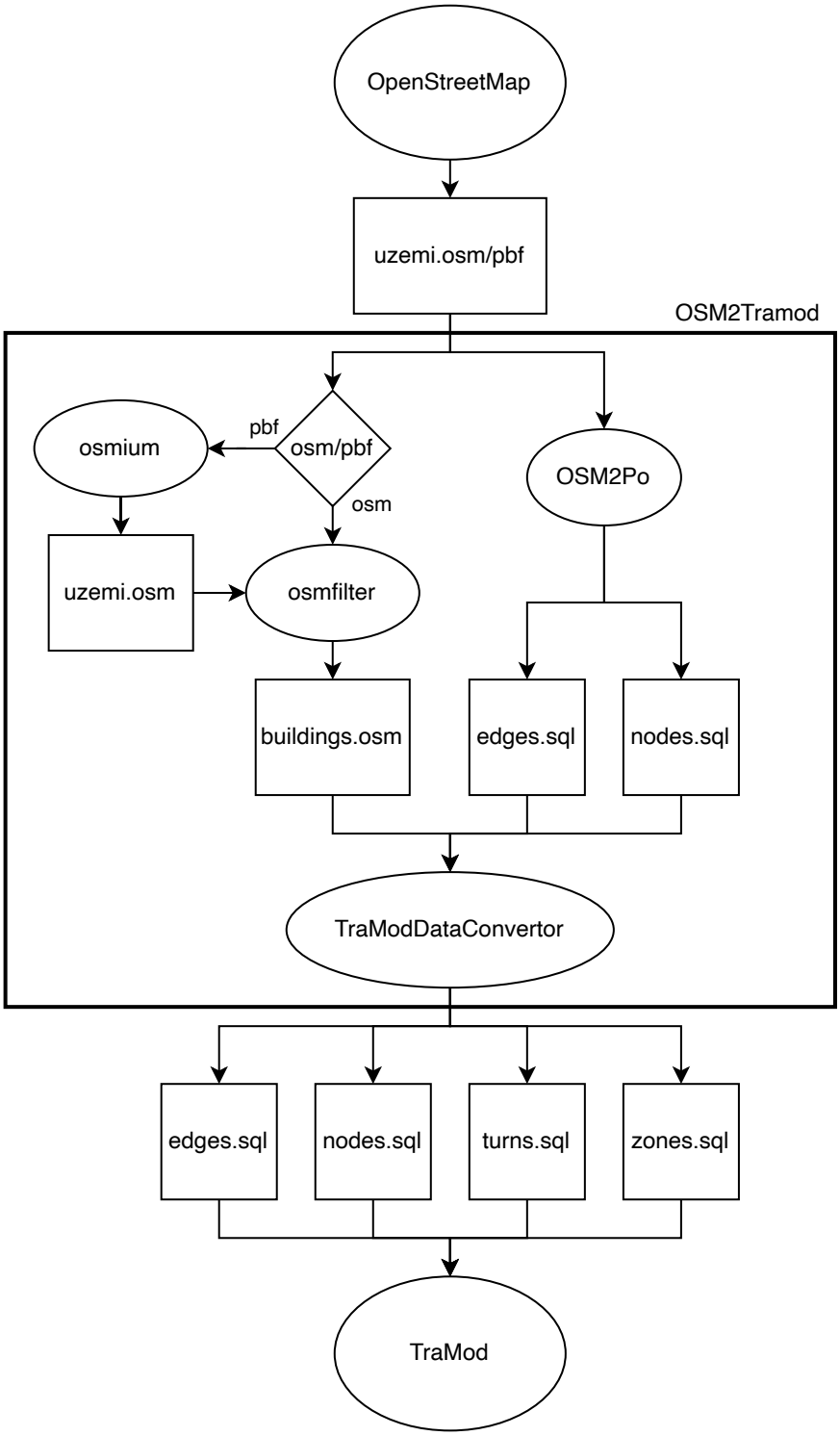
Celkový proces konverze vyžaduje čtyři různé nástroje. Tyto nástroje jsou naprogramované v jazyce Java a jazyce C. K celkovému procesu by uživatel musel všechny tyto nástroje získat, instalovat a poté postupně spouštět. Pro usnadnění uživateli a zaručení kompatibility s operačním systémem byl zvolen software *Docker*. *Docker* je populární nástroj sloužící k vytvoření, nasazení a spouštění kontejnerů. V těchto kontejnerech je uložen software v kompletním souborovém systému, které obsahují vše potřebné k jeho správnému fungování, tedy kód, systémové nástroje, systémové knihovny. Díky tomu bude software vždy běžet stejně bez ohledu na prostředí, ve kterém je nasazen. Nástroj dovoluje aplikacím využívat jádro hostitelského operačního systému a v celém obrazu tedy není celý operační systém. Obsahem obrazu musí být pouze části, které nejsou obsaženy na hostitelském stroji.

Celá aplikace tedy bude dockerizována. Bude obsahovat potřebné externí nástroje, kompilátory a knihovny, které zaručí jejich spustitelnost. Aplikace tak bude spustitelná na jakémkoliv linuxovém stroji, který bude mít nainstalovaný nástroj Docker.

https://web.archive.org/web/20190913100835/http://maureenogara.syscon.com/node/2747331 https://is.muni.cz/th/wu1mq/DP__Ales__Mracko.pdf

5.2.3 Transformace

Připravená data je nyní možné použít pro vlastní transformace, které umožní jejich použití v nástroji Traffic Modeller. V této části jsou jednotlivé transformace popsány. Těmito transformacemi jsou:



Obrázek 5.3: Proces konverze

- tvorba jednosměrných hran z obousměrných
- tvorba zakázaných směrů
- tvorba generátorů dopravy

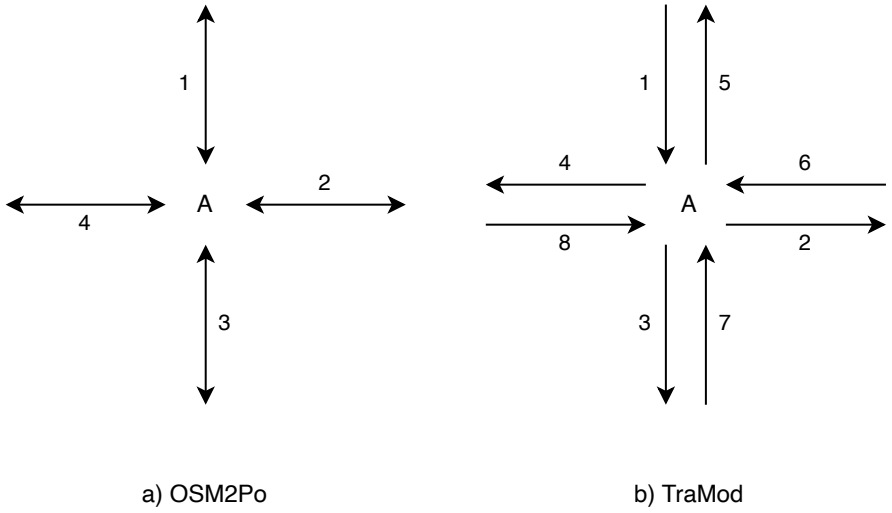
Tvorba jednosměrných hran z obousměrných

První potřebná transformace vychází z reprezentace hran v nástrojích *OSM2Po* a Traffic Modeller. Nástroje *OSM2Po* reprezentuje hrany jako neorientované. Pokud existuje mezi křižovatkami vozovka, po které je možné jet v obou směrech (obousměrná), v datech se vyskytuje pouze jeden záznam - neorientovaná hrana. OSM2Po reprezentuje směr hrany vždy od zdrojového (*source*) k cílovému (*target*) vrcholu. Obousměrnost této hrany je vyjádřena hodnotami sloupce *cost* a *reverse_cost*. Hodnota *cost* je vždy nastavena na hodnotu, která vyjadřuje cenu za přejetí této hrany ve směru hrany, tedy od zdrojového ke koncovému vrcholu. Tato hodnota je vypočítána jako podíl délky této hrany v kilometrech a maximální povolené rychlosti v km/h. Hodnotou je tedy čas potřebný pro přejetí tohoto úseku v hodinách. Obousměrnost je vyjádřena hlavně hodnotou sloupce *revese_cost*. Pokud je silnice jednosměrná, tedy vede jen od zdrojového k cílovému vrcholu, poté je hodnota *reverse_cost* nastavena na hodnotu *1000000*. Pokud je však hrana obousměrná, tedy je možné hranu přejet i ve směru od cílového ke zdrojovému vrcholu, hodnota sloupce *revese_cost* je opět vypočtena. Tímto způsobem OSM2Po reprezentuje neorientované hrany.

Nástroj Traffic Modeller napříč tomu neuchovává hranu jako neorientovanou a v případě obousměrné vozovky vyžaduje zvláštní záznam pro hrany v obou směrech - orientované. První transformací je tedy detekce obousměrných hran a pro všechny vytvořit druhou hranu v opačném směru. Tento jednoduchý proces je možné vidět na obrázku 5.2.. Hrana má totožné hodnoty jako původní hrana. Změnou je pouze výměna zdrojového a cílového vrcholů, pro hodnotu *cost* využita hodnota *reverse_cost* a otočena geometrie pomocí vložení řetězce *ST_Reverse*, což je funkce PostGIS databáze, která při nahrání geometrii otočí na opačný směr.

Tvorba zakázaných směrů

Druhou transformací, která je potřeba aplikovat na připravená data se týká zakázaných směrů. Zakázané směry jsou v datech nástroje OSM2Po uvedeny v tabulce vrcholů ve sloupci *restrictions*. Každý vrchol obsahuje textový záznam o zakázaném směru, které přísluší tomuto vrcholu. Tento textový



Obrázek 5.4: Tvorba obousměrných hran

záznam využívá jednoduchý formát. Zakázaný směr začíná znakem "-", následuje id zdrojové hrany, poté znak "_"a následuje id cílové hrany. Zákaz odbočení je tedy zanesen následovně:

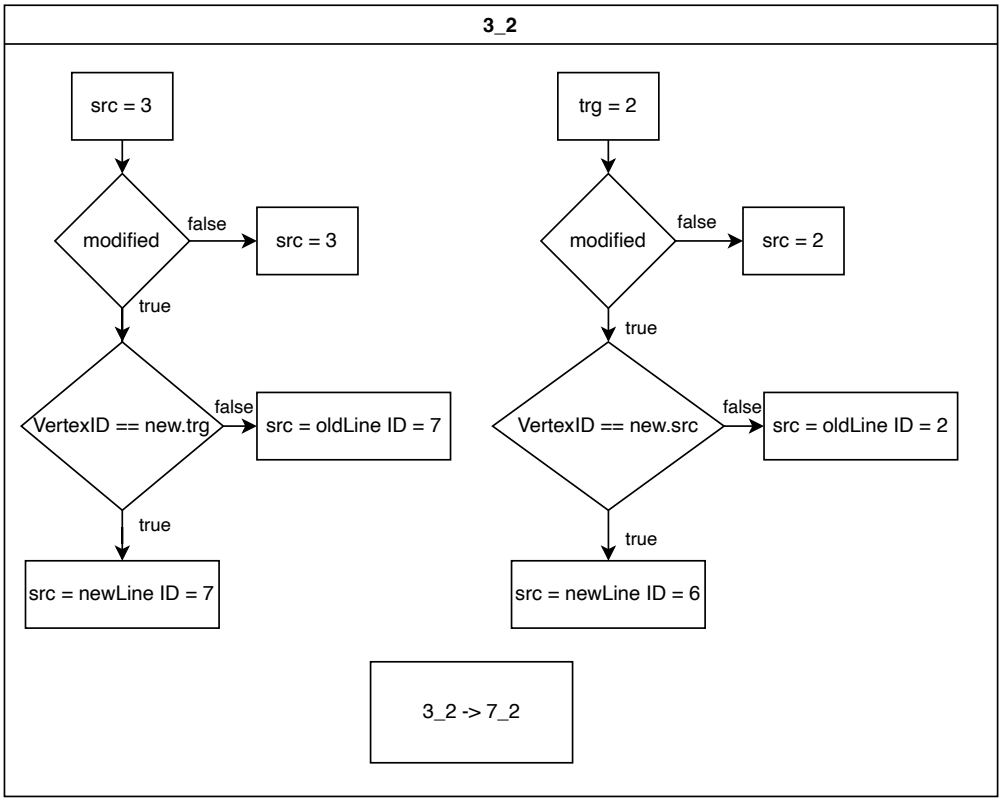
- *-<id_zdrojové_hrany>_<id_cílové_hrany>*

Pokud ve vrcholu existuje více zakázaných směrů, jsou bez jakýchkoliv dalších znaků zřetězené za sebou.

V nástroji Traffic Modeller jsou zakázané směry uloženy v samostatné tabulce. Z každého zakázaného směru, který je uložen v textové podobě ve vrcholu je tedy potřeba vytvořit samostatný záznam do tabulky *turn_restrictions*. Při této transformaci je důležité nezapomenout na předchozí transformaci neorientovaných hran. Pokud byla hrana rozdělena na dvě orientované, je potřeba k tomu přihlédnout i při vytváření záznamů v tabulce *turn_restrictions* a zajistit správné přiřazení identifikátorů hran.

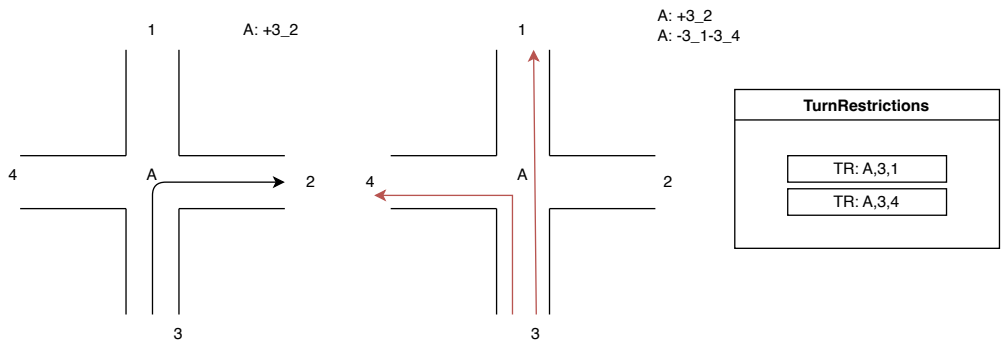
Algoritmus tohoto procesu je vidět na obrázku 4.5. Hodnoty identifikátorů hran odpovídají obrázku 5.2.. Nejdříve je vybrána zdrojová hrana zákazu odbočení. Pokud hrana byla jednosměrná a nebyla tedy vytvořena druhá hrana v opačném směru, zdrojová hrana zákazu odbočení zůstane stejná. Pokud byla hrana obousměrná, je vybrána nová hrana a zkontrolován její cílový vrchol. Pokud nejsou stejné, zamená to, že nová hrana vede z vrcholu ven a je použit indentifikátor původní hrany. Pokud stejné jsou, znamená to, že hraně vedoucí do tohoto vrcholu odpovídá nově vytvořená hrana a je tedy použit její identifikátor. Stejným způsobem je poté vybrána

cílová hrana zákazu odbočení a je zkontrolováno, zda nebyla rozdělena na dvě a pokud ano, je jediným rozdílem oproti algoritmu pro zdrojovou hranu neporovnávání s cílovým vrcholem nově vytvořené hrany, ale se zdrojovým vrcholem nově vytvořené hrany. S přihlédnutím k hodnotám na obrázku 5.2. je tedy vidět, že zákaz odbočení *-3_2* byl modifikován na zákaz odbočení *-7_2*.



Obrázek 5.5: Transformace zákazů odbočení pro obousměrné hrany

V OSM2Po se však vyskytují také případy, kde jsou místo zakázaných směrů použity přikázané směry. Jsou také vypsané v hodnotě *restrictions* u každého vrcholu. Jediným rozdílem v zápisu je využití znaménka + namísto původně použitého -. Jelikož cílová struktura nástroje Traffic Modeller neobsahuje žádný zvláštní způsob pro uložení přikázaných směrů, musí být příkázané směry transformovány na na zakázané všech do hran, které v příkazu uvedené nejsou. Duležité je také opět zachovat správné identifikátory hran v případech, že byly hrany obousměrné a byly tedy rozděleny. Příklad transformace příkazu odbočení na zákazy odbočení je možné vidět na obrázku 4.6..



Obrázek 5.6: Transformace přikázaných směrů

Tímto způsobem je možné transformovat přikázané směry na zakázané směry. V datech se vyskytují i kombinace výše zmíněných transformací (např. 2 přikázané směry). Podrobněji popsané řešení těchto situací je popsáno v kapitole implementace nástroje.

Po aplikaci těchto transformací na výstupní data nástroje OSM2Po odpovídají data modelu silniční sítě v nástroji Traffic Modeller a mohou pro něj být použity.

Tvorba generátorů dopravy

Jak už bylo řečeno, generátory dopravy jsou nezbytnou součástí nástroje Traffic Modeller umožňující nástroji modelovat hustotu dopravy na vymezeném území. Výpočet generátorů dopravy využívá data o budovách. Jednotlivé generátory je poté potřeba shlukovat do menších oblastí a v poslední řadě musí být generátory přiřazeny nejvhodnějšímu uzlu uvnitř oblasti.

Abychom zajistili nejlepší odhad hodnoty generátoru dopravy, je potřeba výpočet měnit pro různé typy budov, počet jejich pater a dalších různých parametrů. Tento výpočet se také liší pro různá území. Výpočet pro území města Plzně je předmětem **bakalářské práce Pavla Blahníka**. Pro naše účely však stačí dokázat, že se generátory dopravy dají z dostupných dat o budovách vypočítat. Hodnotu generátoru dopravy tedy bude reprezentovat plocha budovy a výsledná hodnota tedy bude pouze reprezentativní hodnotou vyjadřující existenci generátoru v dané oblasti. V kapitole popisující datovou strukturu nástroje *OSM* je popis reprezentace nejednobodových prostorových elementů. Tyto elementy jsou popsány sledem několika bodů referencovaných jednou cestou. Záznam o budově je tedy v *OSM* reprezentován jako sled několika bodů, které společně tvoří uzavřený polygon. Tyto

body mají uložené své souřadnice. Díky tomu jsme z polygonu a souřadnic jeho bodů schopni vypočítat plochu budovy. Jelikož jsou budovy tvořeny polygony, jejichž strany se nepřekrývají, je možné pro výpočet plochy použít Gaussovu metodu pro výpočet plochy polygonu. Pro výpočet je tedy možné použít vzorec 5.7.

$$S = \frac{1}{2}\sum_{i=1}^{n-1}(y_i + y_{i+1})(x_i - x_{i+1})$$

Obrázek 5.7: Gaussova metoda pro výpočet plochy polygonu

S označuje plochu polygonu v m^2. Proměnná x vyjadřuje zeměpisnou délku a proměnná y vyjadřuje zeměpisnou šířku. Pro výpočet v m^2 jsou souřadnice převedeny ze souřadnicového systému WGS-84 do projekce Web Mercator, která je na metrech založena. K převodu je použit vzorec 5.8. Podmínkou pro správné fungování vzorce 5.7 je uvedení prvního vrcholu polygonu i jako posledního. Tyto generátory je dále potřeba shlukovat do nějakých menších oblastí.

$$x = \lambda * 20037508.34/180$$
$$y = (\log(\tan((90 + \varphi) * \pi/360))/(\pi/180)) * 20037508.34/180$$

Obrázek 5.8: Projekce WGS-84 na Web Mercator

φ souřadnice centroidu C vyjadřující zeměpisnou šířku a C_λ je λ souřadnice centroidu C vyjadřující zeměpisnou délku.

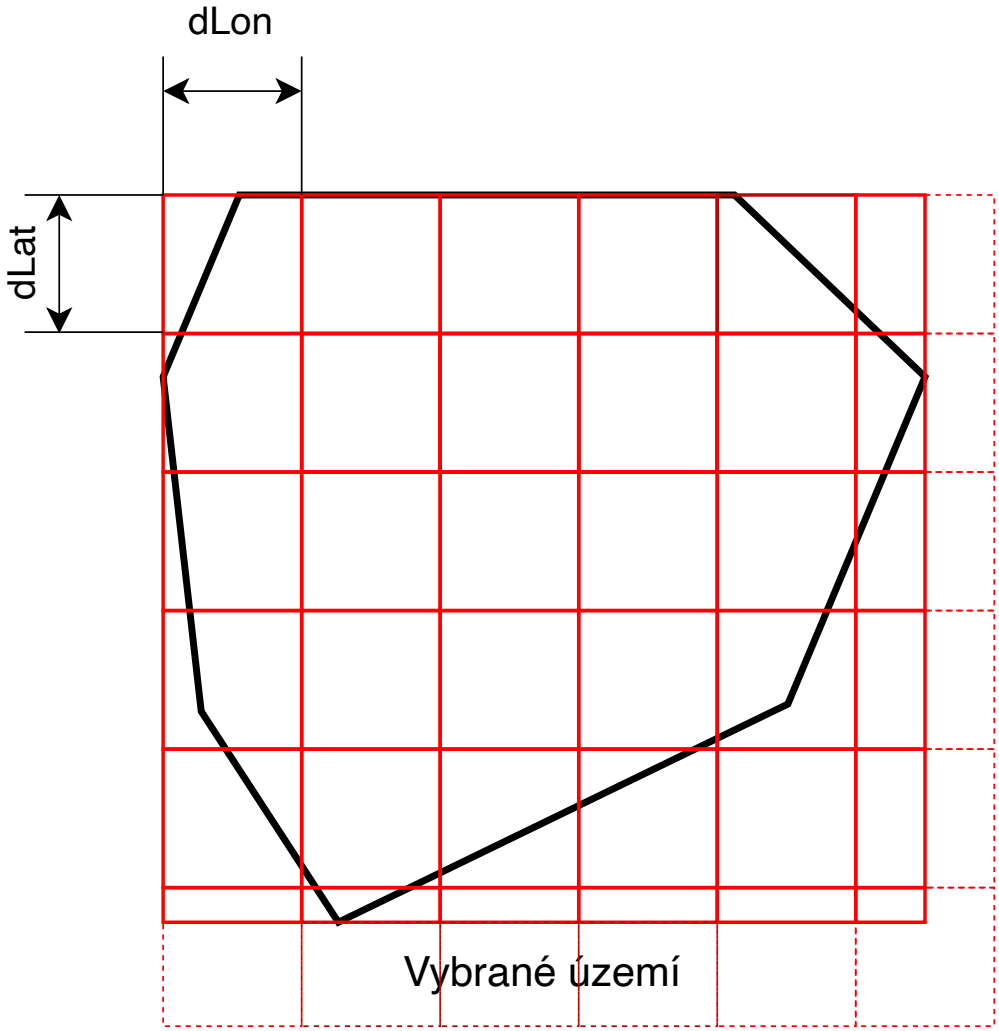
Vymezení vhodných oblastí opět závisí na několika různých faktorech, jako může být členění městských částí, přítomnost řek a další. Tento výpočet opět není předmětem diplomové práce a pro její účely postačí vyznačené území rozdělit na libovolné oblasti. Pro účely diplomové práce je voleno dělení požadované oblasti na čtvercovou mřížku. Počet čtverců, na které je oblast rozdělena volí uživatel. Délka strany jednoho čtverce je poté vypočítána podle následujícího vzorce 5.9. **https://gist.github.com/onderaltintas/6649521**

$$a = \sqrt{\frac{S}{c}}$$

Obrázek 5.9: Výpočet délky strany dlaždice

Délka strany a je vypočtena jako druhá odmocnina z podílu plochy S a uživatelem zadaného požadovaného počtu dlaždic c. Plocha S je vypočítána také podle vzorce 5.7, ale jejím vstupem jsou souřadnice ve formátu WGS-84, tím pádem se jedná o plochu ve stupních. Vzhledem k libovolnému poměru

délky stran celé oblasti, není možné ji rozdělit na čtvercovou mřížku tak, aby byla vyplněna kompletně. Čtverce tedy budou tvořeny od jednoho rohu a poslední oblastí nebude čtverec, ale pouze oblast s délkou strany takovou, aby vyplnila zbylou mezeru. Poslední krajní oblasti jsou tedy pouze malou doplňkovou oblastí, aby byla celá oblast pokryta. Grafické znázornění je vidět na obrázku:



Obrázek 5.10: Vytvoření oblastí

Délka strany a je na obrázku znázorněna pomocí proměnných $dLon$ a $dLat$. Algoritmus postupně tvoří dlaždice a porovnává hodnoty nejvyšší hodnoty zeměpisné šířky a délky dlaždice s hodnotami zeměpisné šířky a délky celé vymezené oblasti. Pokud je nejvyšší hodnota šířky nebo délky vyšší než

nejvyšší hodnota šířky nebo délky celé oblasti, je pro dlaždici použita hodnota celé oblasti. Tímto způsobem je vytvořena mřížka pokrývající celou oblast.

V každé oblasti (dlaždici) jsou sečteny příslušné hodnoty generátorů. K příslušnosti budovy do jednotlivé oblasti je využit jejich centroid, aby se zabránilo situacím, kde budova svou plochou zasahuje do více než jedné oblasti. Centroid budov je vypočten vzorcem na obrázku 5.6.

$$C_\lambda = \frac{1}{6S}\sum_{i=0}^{n-1}(\lambda_i + \lambda_{i+1})(\varphi_i\lambda_{i+1} - \varphi_{i+1}\lambda_i)$$
$$C_\varphi = \frac{1}{6S}\sum_{i=0}^{n-1}(\varphi_i + \varphi_{i+1})(\varphi_i\lambda_{i+1} - \varphi_{i+1}\lambda_i)$$

Obrázek 5.11: Výpočet centroidů polygonu

C_φ je φ souřadnice centroidu C vyjadřující zeměpisnou šířku a C_λ je λ souřadnice centroidu C vyjadřující zeměpisnou délku. $N-1$ je počet vrcholů (první musí být uveden i jako poslední). φ a λ jsou souřadnice jednotlivých bodů polygonu. **BOURKE http://paulbourke.net/geometry/polygonmesh/**

Generátor dopravy je v poslední řadě potřeba přiřadit nějaké vhodné křižovatce v dopravní síti. Výběr vhodné křižovatky, ke které je generátor přiřazen, je opět důležitým faktorem správné simulace dopravní situace. Tento výběr také není součástí diplomové práce a pro její účely bude generátor přiřazen křižovatce, která je nejblíže středu dlaždice. K výpočtu vzdálenosti mezi dvěma body (centroidu dlaždice a křižovatky) je možné využít Haversinův vzorec uvedený na obrázku 5.7.

$$a = \sin^2(\Delta\varphi/2) + \cos\varphi_1 \cdot \cos\varphi_2 \cdot \sin^2(\Delta\lambda/2)$$
$$c = 2 \cdot atan2(\sqrt{a}, \sqrt{(1-a)})$$
$$d = R \cdot c$$

Obrázek 5.12: Výpočet vzdálenosti dvou bodů

φ je zeměpisná šířka, λ je zeměpisná délka, R je průměr Země (6371m). **HAVERSINE - https://www.movable-type.co.uk/scripts/latlong.html**

Generátory jsou tedy přiřazeny křižovatce ležící nejblíže středu dlaždice. Pokud však dlaždice obsahuje generátor a neobsahuje žádnou křižovatku, je hodnota generátoru připočtena sousedící dlaždici s nejvyšší hodnotou generátoru obsahující křižovatku. Pokud žádná sousední dlaždice neobsahuje křižovatku, znamená to, že uživatel zvolil příliš jemné dělení prostoru a je tedy třeba zvolit méně jemné dělení.

Tímto způsobem jsou vypočteny a přiřazeny generátory dopravy.

5.3 Návrh zpětného algoritmu

Nástroj Traffic Modeller během simulace vypočítá hodnoty intenzity dopravy pro jednotlivé hrany. Intenzita dopravy značí počet vozidel, které přes danou hranu projedou ve špičkové hodině (typicky 4.-5. hodina odpoledne). O tyto hodnoty je možné rozšířit původní data OSM. Tato data však nejsou v současné době nástrojem Traffic Modeller nijak uložena ani poskytována a implementace zpětného rozšíření dat OSM tedy není možná. Následující popis je návrh, který bude možné implementovat, pokud nástroj Traffic Modeller umožní tato data získat přes své API.

Přidání dat hranám v původních datech je možné dvěma způsoby.

1. Přidat značku s hodnotou intenzity dopravy příslušné cestě
2. Vytvořit novou relaci referencovanou na vrcholy v původních datech

Pro využití prvního zmíněného způsobu je nutné, aby jednotlivé hrany v nástroji TraMod odpovídaly cestám v datech OSM. Z této kapitoly však jasně vyplývá, že záznamy hran v nástroji TraMod původní cesty v datech OSM nereferencují. Tento přístup tedy není možné použít.

Pro využití druhého způsobu je potřeba, aby jednotlivé hrany v nástroji TraMod referencovali vrcholy v původních datech OSM. Z popisu datové struktury nástroje však víme, že ani jednotlivé body nejsou hranami referencovány. Z kapitoly 4.2.2 však víme, že jednotlivé vrcholy v datech OSM jsou současně ke křižovatkám použity k určení tvaru jednotlivých cest. Tento tvar je v datové struktuře nástroje TraMod reprezentován hodnotami geometrie. Geometrie jednotlivých hran obsahuje seznam souřadnic bodů, které je tvoří. Získání identifikátorů jednotlivých vrcholů z dat OSM je tedy možné získat postupným prohledáváním jejich souřadnic.

Nástroj vykonávající zpětný algoritmus tedy musí načíst vrcholy z původních dat OSM současně s hranami nástroje TraMod. Pro každou hranu v nástroji TraMod načte seznam jednotlivých souřadnic, které definují její tvar. Pro každou souřadnici je poté potřeba prohledat seznam načtených vrcholů z dat OSM a podle souřadnic nalézt odpovídající vrchol. Po nalezení všech vrcholů je poté pro každou hranu vytvořena nová relace v datech OSM, která referencuje nalezené body ve stejném pořadí, jako jsou uvedeny v geometrii hrany. Do této relace je uložena značka vyjadřující intenzitu dopravy z nástroje TraMod.

Jednotlivé relace poté můžou být připsány do souboru s daty OSM, čímž původní dataset obohatí o hodnoty intenzity dopravy na jednotlivých úsecích silniční sítě.

6 Implementace

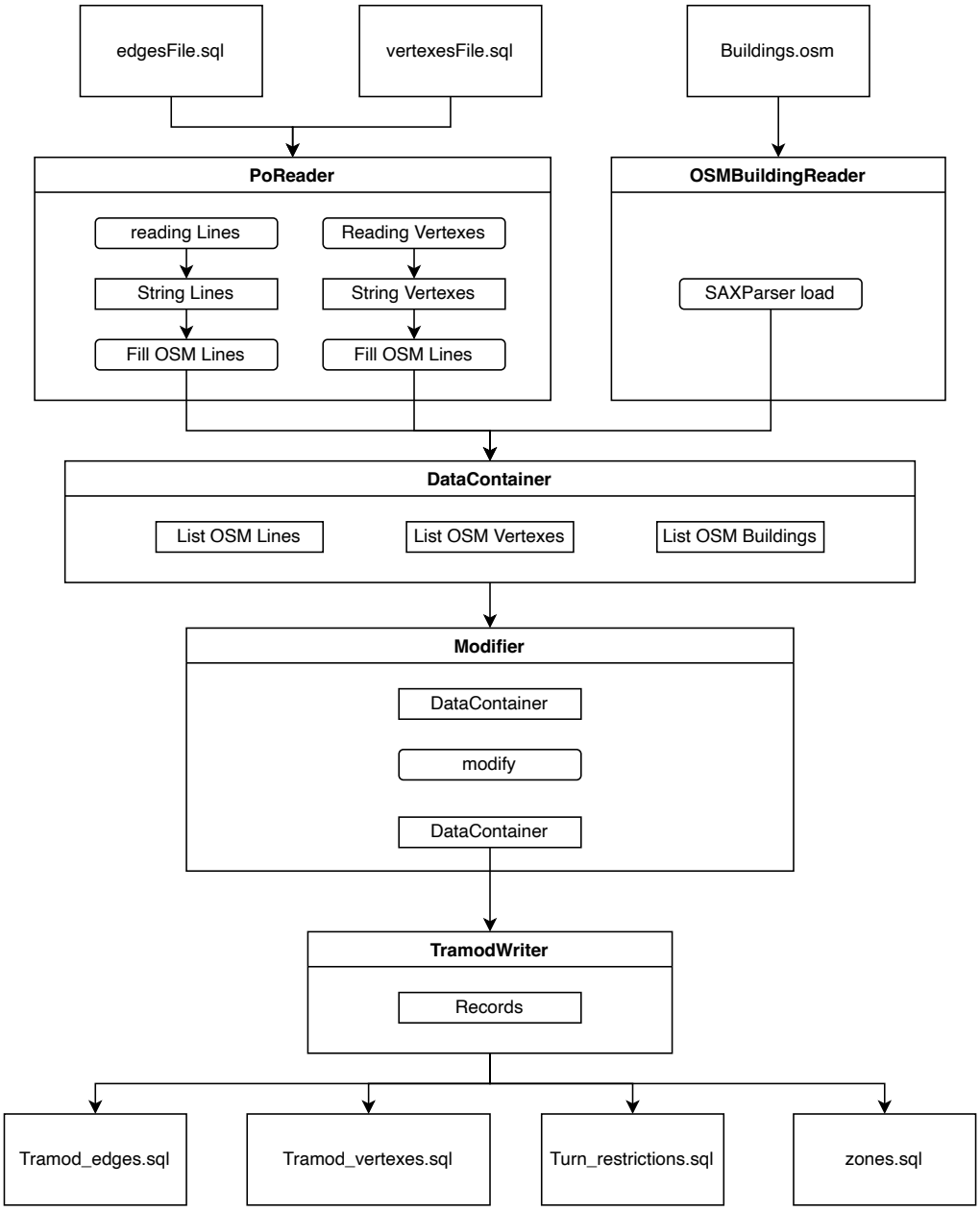
V této kapitole jsou popsány důležité dílčí části implementace nástroje. Vstupem vlastního nástroje jsou výstupní soubory nástroje OSM2Po, které jsou využity k transformaci silniční sítě a výstupní soubor nástroje *osmfilter*, který obsahuje data týkající se budov. Výstupem jsou poté *SQL Dump* soubory, pomocí kterých lze načíst data do databáze nástroje Traffic Modeller. Celkový zjednodušený proces je vidět na obrázku 5.1.

Nástroj nejdříve načte a zpracuje vstupní sql soubory týkající se silniční sítě, následně soubor *osm* s informacemi o budovách. Z těchto souborů naplní vlastní struktury uvnitř třídy *DataContainer*. Poté jsou spuštěny transformační procesy pomocí třídy *Modifier*, která převezme obsah třídy *DataContainer* a postupně zpracuje její záznamy. Výsledky uložené ve třídě *DataContainer* jsou poté pomocí třídy *TramodWriter* uloženy do čtyř *SQL Dump* souborů. Tyto čtyři soubory obsahují příkazy v jazyce SQL a je možné je použít pro nahrání záznamů do databáze nástroje TrafficModeller.

6.1 Import dat

Tato část popisuje načtení dat do nástroje. Vstupními soubory nástroje jsou výstupní *SQL Dump* soubory nástroje *OSM2Po* a výstupní soubor nástroje *osmfilter* ve formátu *osm*. Všechna data jsou v nástroji uložena ve třídě *DataContainer*. Ta obsahuje šest seznamů, které jsou v průběhu aplikace postupně plněny. Těmito seznamy jsou:

- *Lines* - seznam hran načtených z OSM2Po
- *Vertexes* - seznam vrcholů načtených z OSM2Po
- *ModifiedLines* - seznam obsahující kontejner *LineChangesTree*, který slouží k modifikaci potřebných hran
- *TurnRestrictions* - seznam zákazů odbočení
- *nodes* - seznam bodů, jejichž sledy reprezentují budovy
- *buildings* - seznam budov



Obrázek 6.1: Proces transformace vlastním nástrojem

6.1.1 Import dopravní sítě

O načítání souborů s daty o dopravní síti se stará třída *PoReader*. Do konstruktoru jsou předány názvy souborů a třída si inicializuje potřebné atributy. Následně jsou zavolány metody *executeReadingLines()* pro načtení hran do paměti a *executeReadingVertexes()* pro načtení vrcholů do paměti. Metoda *executeReadingLines()* čte soubor s hranami řádek po řádku a rozhoduje, zda načtený řádek odpovídá reprezentaci jedné hrany. O rozhodnutí, zda textová reprezentace jednoho řádku odpovídá záznamu o hraně rozhoduje metoda *isLine()*. Ta ke svému rozhodování využívá regulární výraz:

```
regex = "[(].+[)],?;?"
```

Pokud řádek odpovídá regulárnímu výrazu, uloží jej do seznamu *readLines*. Pokud neodpovídá, uložen není. Každý záznam v seznamu *readLines* tedy odpovídá řetězci obsahující záznam s hranou. V tomto seznamu jsou tedy nadále uložené pouze řádky, které reprezentují právě jednou hranu. Následně je zavolána metoda *fillLines()*, která iteruje přes seznam *readLines* a pomocí metod třídy *String* zpracuje jednotlivé řádky a vytvoří z nich objekty typu *Line* a uloží je do seznamu *Lines* uvnitř třídy *DataContainer*.

Stejným způsobem jsou zpracovány jednotlivé vrcholy. Třída *PoReader* čte pomocí metody *executeReadingVertexes()* soubor s vrcholy řádek po řádku a obdobně jako u hran rozhoduje, zda je načtený řádek reprezentací jednoho vrcholu. Rozhodování opět probíhá pomocí regulárního výrazu:

```
regex = "[(].+[)],?;?"
```

Pokud řádek odpovídá regulárnímu výrazu, uloží jej do seznamu *readVertexes*. Pokud neodpovídá, uložen není. Stejně jako u seznamu *readLines*, každý záznam v tomto seznamu obsahuje textovou reprezentaci právě jednoho vrcholu. Následně je zavolána metoda *fillVertexes()*, která iteruje přes seznam *readVertexes* a pomocí metod třídy *String* zpracuje jednotlivé řádky a vytvoří z nich objekty typu *Vertex*. Tyto záznamy jsou uloženy do seznamu *Vertexes* uvnitř třídy *DataContainer*.

Po načtení jsou ve třídě *DataContainer* naplněny seznamy *Lines* a *Vertexes*.

6.1.2 Import záznamů budov

Narozdíl od dopravní sítě, není soubor s daty o budovách ve formátu *SQL* příkazů. Tento soubor je ve formátu *osm* a je formátovaný pomocí jazyka *XML*. K načtení budov tedy není vytvořen vlastní parser, ale je použita knihovna *SAXParser*. Té je předána třída *DataContainerHandler*, která obsahuje po-

pis významu jednotlivých značek uvnitř souboru. SAXParser následně pomocí metody *load()* čte soubor s daty o budovách a na základě popisu ve třídě *DataContainerHandler* plní třídu *DataContainer*. Každý uzel je reprezentován třídou *OSMNode* a uložen do rozptylové tabulky *nodes* jejímž klíčem je identifikátor uzlu a hodnotou poté samotná instance uzlu. Budovy jsou reprezentovány třídou *OSMBuilding* a jsou uloženy do seznamu *buildings*.

Po importu záznamu budov jsou ve třídě *DataContainer* naplněny seznamy *Nodes*

6.2 Transformace

O transformaci dat se stará třída *Modifier*. Třídě je v konstrukotru předána instance třídy *DataContainer* obsahující načtená data. Následně je spuštěna metoda *execute()*. Tato metoda postupně spouští veškeré procesy, které data transformují. V této části jsou popsány jednotlivé procesy.

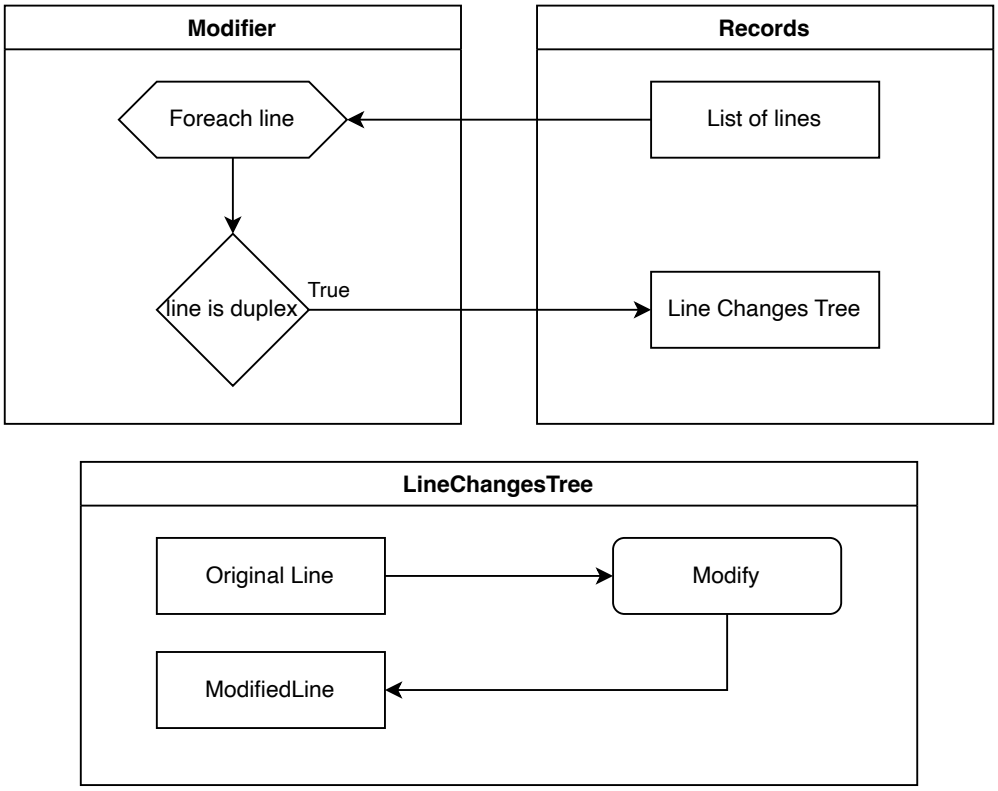
6.2.1 Obousměrné hrany

O první transformaci dat se stará metoda *processLines()*. Tato metoda slouží k vytvoření duplicitní hrany v opačném směru pro každou načtenou obousměrnou hranu. Metoda iteruje přes seznam všechn hran načtených do paměti a detekuje obousměrné hrany pomocí booleanovské metody *isDuplex()*. Metoda se rozhoduje podle hodnoty proměnné *reverse_cost*. Pokud je tato hodnota *1000000.0*, není hrana obousměrná a metoda vrací hodnotu *false*. Pokud hodnota proměnné *reverse_cost* není *1000000.0*, je hrana obousměrná a metoda vrací hodnotu *true*. V případě obousměrné hrany vytvoří záznam v hashovací tabulce *modifiedLines*. Klíčem záznamu v tabulce je identifikátor hrany a hodnotou poté instance třídy *LineChangesTree*. Tato třída slouží jako kontejner obousměrných hran, pro které je třeba vytvořit hranu v opačném směru. Obsahuje referenci na originální hranu a současně nově vytvořenou hranu v opačném směru.

Po nalezení všech obousměrných hran je potřeba najít počáteční identifikátor nově tvořených hran. Počáteční identifikátor je jednoduše získán jako velikost listu *lines*. Data z nástroje OSM2Po pro n hran nabývají hodnot *{1, 2, ..., n}*, nejsou v nich žádné mezery. Pro m nově vytvořených hran budou jejich identifikátory nabývat hodnot *{n+1, n+2, ... ,n+m}*. Po nalezení počátečního identifikátoru nově tvořených hran je iterováno přes hashovací tabulku *modifiedLines* a pro každý záznam je volána metoda *modify()* třídy *LineChangesTree*. Tato metoda vytvoří novou hranu, které nastaví nový identifikátor, zamění hodnoty zdrojového a cílového vrcholu a převrátí

geometrii. Geometrie je převrácena připsáním řetězce *ST_REVERSE* před hodnotu geometrie.

Proces zpracování hran je vidět na obrázku 5.2.



Obrázek 6.2: Zpracování hran

6.2.2 Zakázané směry

Dalším procesem spuštěným třídou *Modifier* je zpracování vrcholů. Při zpracování vrcholů jsou vytvořeny záznamy zakázaných směrů, které jsou uloženy v atributu *restrictions* vrcholu. Zákázané směry jsou uloženy do listu *TurnRestrictions* uvnitř třídy *DataContainer*.

Prvním krokem je vytvoření zakázaných směrů z přikázaných směrů. Tento proces je znázorněn na obrázku 4.6.. Program nejdříve vytvoří dvě kopie původního seznamu zákazů a příkazů. Jedna kopie bude sloužit k porovnávání nově vytvořených zákazů s původními příkazy a v druhé kopii budou postupně zaměňovány příkazy za zákazy. Jednotlivé příkazy odbočení

jsou ve svém textovém zápisu uloženy do seznamu. Pro každý příkaz odbočení ze seznamu je poté spuštěna metoda *orderToRestriction()*. Vstupními parametry této funkce jsou vrchol obsahující tento příkaz, samotný příkaz v textové podobě a kopie *restrictions*. Výstupem této funkce jsou nově vytvořené zákazy v textové podobě. Metoda ponechá zdrojovou hranu a poté hledá všechny hrany v seznamu původních hran, jejichž zdrojový nebo cílový vrchol je právě vrchol předaný metodě. Důvodem testování zdrojového i cílového vrcholu hrany je prohledávání seznamu všech původních hran (obousměrných i jednosměrných). Může nastat případ, kdy hrana byla obousměrná a tudíž bude v pozdější části zpracování použit nově vytvořený identifikátor pro otočenou hranu. Je-li tedy příkaz ve vrcholu A zapsán jako *+x_y*, jsou nalezeny všechny hrany, pro které platí *src*=A, nebo *trg*=A. Identifikátor této hrany (z) je poté zaměněn s identifikátoremm hrany y a vznikne tedy nový zákaz *-x_z*. Tento zákaz je porovnán s kopií původního textového záznamu *restrictions*, zda neexistuje jiný příkaz, který by s tímto zákazem byl totožný. Pokud nalezen není, je zákaz připojen k textové reprezentaci nově tvořených zákazů. Po nalezení všech hran proces končí a metoda vrátí zpět nově vytvořené zákazy a nahradí jimi příslušný původní příkaz.

Dalším krok je záměna identifikátorů hran z důvodu transformace obousměrných hran. Pro každý zákaz je tedy spuštěn algoritmus zobrazený na obrázku 4.5.. O tento proces se stará metoda *turnRestricionSwitch()*, které je parametrem předán příslušný vrchol a textová reprezentace jednoho zákazu. Metoda zjistí, zda v seznamu modifikovaných hran existuje hrana se stejným identifikátorem. Pokud hranu nenalezne, není identifikátor zdrojové hrany dále zpracováván. Pokud hranu nalezne, je využita třída *LineChangesTree* a je detekováno, zda tento vrchol referencuje původní, či nově vytvořená hrana jako svůj cílový vrchol. Identifikátor odpovídající hrany je potom uložen jako zdrojová hrana tohoto zákazu. Stejným způsobem jsou zpracovány cílové hrany zákazu. Jediným rozdílem je detekce reference vrcholu jako zdrojové hrany místo cílové.

Posledním krokem je z jednotlivých zákazů vytvořit samostatné záznamy v seznamu *TurnRestrictions* ve třídě *DataContainer*. Textový řetězec s validně modifikovanými zákazy může obsahovat duplicity. Tyto duplicity mohly vzniknout v případě, že v původním řetězci existoval více než jeden příkaz odbočení. V takovém případě mohly být některé příkazy nahrazeny stejnými zákazy. Před uložením nového zákazu odbočení do seznamu je tedy testováno, zda konkrétní zákaz není již v seznamu uložen. Pokud ne, je vytvořen nový záznam, který je uložen do seznamu ve třídě *DataContainer*.

https://github.com/locationtech/jts.

6.2.3 Výpočet generátorů dopravy

V části *Tvorba generátorů dopravy* kapitoly 4.2.2. je popsán způsob výpočtu generátorů dopravy. Tyto generátory jsou počítány ze záznamů budov. Výpočet a přiřazení těchto generátorů křižovatkám probíhá v několika procesech pomocí několika metod, které jsou v této části popsány.

Převod geometrie

Prvním procesem je příprava geometrie vrcholů reprezentujících křižovatky. Nástroj OSM2Po využívá hexadecimální zápis geometrie a ten je také načten vlastním nástrojem. Pro přiřazení generátorů křižovatkám je však nezbytné, aby bylo možné s prostorovými daty počítat. Je tedy potřeba tento hexadecimální zápis převést na hodnoty zeměpisné šířky a zeměpisné délky. O to se stará metoda *convertToCoordinates()*. Tato metoda využívá *JTS*. *JTS* je knihovna v jazycce Java umožňující manipulaci s vektorovou gemoetrií. Pomocí této knihovny je uvnitř metody převeden hexadecimální zápis geometrie vrcholu na hodnoty zeměpisné šířky a délky. Takto zaznamenaná geometrie umožňuje další výpočty, které jsou součástí nástroje.

https://github.com/locationtech/jts.

Výpočet jednotlivých generátorů

Dalším procesem je samotný výpočet generátorů. Generátory jsou počítány z budov podle vzorce 5.7. Výpočet je volán metodou *calculateBuildingAreas()*, která iteruje přes seznam načtených budov a pro každou spouští metodu *calculateArea()*, která pomocí uvedeného vzorce vypočítá plochu budovy. Výpočet využívá souřadnic jednotlivých bodů polygonu, který reprezentuje budovu. Tyto souřadnice jsou však načteny ve stupních. Aby výsledná hodnota reprezentovala plochu v m^2, je třeba souřadnice převést na metry. O tento převod se stará metoda *degreesToMeters()*.

https://stackoverflow.com/questions/639695/how-to-convert-latitude-or-longitude-to-meters

Pro přiřazení generátorů správné oblasti je nutné jednoznačně určit, ve které oblasti se budova nachází. Jelikož by plocha budovy mohla zasahovat do více oblastí, k jednoznačnému určení se tedy použije centroid budovy. Vzorec pro výpočet centroidu budovy je uveden v kapitole 4.2.2.. Výpočet je volán metodou *calculateBuildingCentroids()*, která pro každou budovu v seznamu volá metodu *calculateCentroid()*. Narozdíl od plochy, u které bylo potřeba převést souřadnice ze stupňu na metry, souřadnice centroidu jsou také ve stupních. Pro výpočet centroidu je také nutné vypočítat plo-

chu budovy. Abychom však byla zachována konzistence jednotek, výpočet nepoužije již vypočítanou plochu budovy v metrech, avšak vypočítá plochu znovu s hodnotami souřadnic ve stupních. Tento výpočet je zařízen metodou *calculateSphericArea()*. Tímto způsobem jsou tedy vypočítány jednotlivé generátory a centroidy pro každou zaznamenanou budovu.

Přiřazení generátorů

K přiřazení generátorů je nejdříve potřeba inicializovat jednotlivé oblasti. O to se stará metoda *processGrid()*. Metoda nejdříve spustí inicializaci třídy *Grid* uložené ve třídě *DataContainer*. Během inicializace mřížky je nastaven počet čtverců požadovaný uživatelem a pomocí hranic oblasti uložené ve třídě *OSMBounds* vypočteny přírustky zeměpisné šířky a zeměpisné délky tak, jak je popsáno v kapitole 4.2.2. v části *Tvorba generátorů dopravy*. V poslední řadě je inicializována hashovací tabulka, jejíž klíčem je třída *GridLocator* a hodnotou třída *Tile* reprezentující jednu dlaždici. Samotné instance jednotlivých dlaždic v tuto chvíli vytvořeny nejsou.

Metoda *processGrid()* po inicializaci spustí metodu *calculateGenerators()* třídy *Grid*. Metoda iteruje přes seznam všech načtených budov. U každé budovy je nejdříve zkontrolováno, zda je vypočtený centroid budovy uvnitř hranic celé oblasti. Následně je pomocí přírustků detekována dlaždice, ve které se budova nachází. K detekování dlaždice se využívá právě třída *GridLocator*. Třída *GridLocator* reprezentuje pomyslnou souřadnici dlaždice. Má uložené dva atributy, které reprezentují pořadí dlaždice v horizontálním a vertikálním směru. Pro budovu je tedy nalezena souřadnice dlaždice, ve které se nachází. Pokud v hashovací tabulce ještě neexistuje záznam na této souřadnici, je dlaždice vytvořena a je v ní nastavena hodnota generátoru budovy. Pokud již v hashovací tabulce záznam na této souřadnici existuje, je generátor z budovy přičten k již existující hodnotě generátoru uvnitř dlaždice. Tímto způsobem je zařízeno, že jsou v paměti vytvořeny pouze dlaždice tam, kde se vyskytují generátory vyskytují.

Další částí je přiřazení generátorů křižovatkám. Podle kapitoly 4.2.2 jsou generátory přiřazeny křižovatce, která je nejblíže středu (centroidu) dlaždice. O přiřazení generátorů se stará metoda *assignTrafficGenerators()*. Tato metoda iteruje přes seznam načtených křižovatek. Pro každou křižovatku zjistí souřadnici dlaždice, ve které se nachází, stejným způsobem jako metoda *calculateGenerators* pro budovy. Opět je testováno, zda v hashovací tabulce mřížky existuje dlaždice s touto souřadnicí (v hashovací tabulce se vyskytuje záznam s tímto klíčem). Pokud dlaždice neexistuje, je vytvořena a křižovatka dlaždici nastavena. Pokud existuje, je pomocí booleanovské metody *isClo-*

ser() detekováno, zda je blíže centroidu dlaždice, než dosud nejbližší křižovatka. Pokud ano, je tato křižovatka mřížce nastavena jako dosud nejbližší. Po skončení metody jsou dlaždicím přiřazeny křižovatky nejbližší jejich centroidům.

Posledním procesem je ošetření situací, kdy se v dlaždici vyskytují budovy a má tedy generátor dopravy, ale zároveň neobsahuje žádnou křižovatku. O toto přiřazení se stará metoda *assignVertexlessGenerators()*. Tato metoda s využitím třídy *GridLocator* držící souřadnice dlaždic prohledá okolní dlaždice a najde takovou, která má největší hodnotu generátoru. K takové dlaždici je poté generátor dopravy přičten a je přiřazen k příslušné křižovatce. Pokud v hashovací tabulce není vytvořena žádná taková dlaždice, nebo nelze nalézt dlaždici, která obsahuje křižovatku, je uživateli vypsána hláška, že zvolil příliš jemné dělení oblasti a tento generátor není přiřazen.

V poslední řadě je znovu iterováno přes vytvořené dlaždice a každé je přiřazen identifikátor pro účely importu do databáze. To má na starost metoda *assignZoneIDs()*.

6.3 Export dat

Výstupem nástroje jsou čtyři SQL Dump soubory, pomocí kterých lze nahrát data do databáze nástroje Traffic Modeller. O vytvoření těchto souborů se stará třída *TramodWriter*. Třída obsahuje tyto metody, z nichž každá vytvoří příslušný soubor a naplní jej daty. Těmito metodami jsou:

- *executeWritingScheme()* - vytvoří soubor, jenž vytvoří tabulky v databázi
- *executeWritingLines()* - vytvoří soubor s hranami reprezentující vozovky
- *executeWritingTurnRestrictions()* vytvoří soubor se zakázanými směry
- *executeWritingVertexes()* - vytvoří soubor s vrcholy reprezentující křižovatky
- *executeWritingZones()* - vytvoří soubor s generátory dopravy

Každá metoda prochází příslušný seznam uložený ve třídě *DataContainer*. Všechny třídy reprezentující příslušné elementy mají implementovanou metodu *toSQLString()*, která příslušná data vypíše v syntaxi jazyka SQL. V každé metodě je tedy iterováno přes seznamy elementů a pro každý element volána tato metoda. Do všech souborů je vypisováno pomocí třídy *FileWriter* knihovny *java.io*

6.4 Docker

Aplikace je dockerizována, což umožňuje její nasazení v serverovém prostředí. Je zabalena do kontejneru, což je zapouzdřené prostředí, které obsahuje pouze požadované nástroje a pro ně specifické soubory a knihovny. Těmito nástroji jsou popisované nástroje *OSM2Po*, *osmium*, *osmfilter* a vlastní nástroj. Nástroj *OSM2Po* je naprogramovaný v jazyce Java a je spustitelný přes přiložený *JAR* soubor. Současně musí být dockeru předány nástroje pro práci s Javou, tedy JDK. Nástroj *osmium* je součástí klasických repozitářů systému Ubuntu a je tedy možné jej v dockeru nainstalovat pomocí package manageru. Nástroj *osmfilter* je napsaný v jazyce *C*. Zdrojový soubor je také potřeba přeložit a dockeru nainstalovat a proto musí být dockeru předán kompilátor jazyka *C* - *gcc*. V poslední řadě vlastní nástroj je také naprogramován v Javě a je sestaven pomocí software *Apache Maven*.

Konfigurace, pomocí které je aplikace dockerizována, je uložena v souboru *Dockerfile*. Bázovým obrazem (z angl. *base image*) , který je v Dockeru využit je *Ubuntu:jammy*. Pomocí package manageru tohoto systému *apt-get* jsou nainstalovány potřebné nástroje a knihovny. Těmi jsou:

- *gcc* - překladač jazyka *C*, pro nástroj *osmfilter*
- *openjdk 8* - java pro nástroje osm2po a nástroj vytvořený v rámci projektu
- *Apache Maven* - nástroj sestavení vytvořené aplikace
- *Osmium* - Nástroj ke konverzi formátu souboru s daty OSM

V souboru *Dockerfile* je dále popsáno kopírování souborů a kompilace jednotlivých aplikací. Na konci je spuštěn shell script, který vykonává proces na obrázku 4.3.

Samotná dockerizace a spuštění je provedena pomocí samostatného shell-scriptu, který zkontroluje počet vstupních parametrů, nastaví dockeru proměnné a spustí dockerizaci pomocí nástroje *docker-compose*. Po skončení dockerizace a běhu všech nástrojů jsou ve výstupní složce definované uživatelem k nalezení výstupní soubory nástroje.

7 Uživatelská příručka

V této kapitole jsou popsány informace pro uživatele nástroje. Je zde příklad stažení požadovaných dat, návod na spuštění aplikace včetně popisu vstupních parametrů a také způsob importu dat do databáze nástroje TraMod.

7.1 Stažení dat

Data je možné získat z více zdrojů různých zdrojů. Popsány zde budou dva příklady a to stažení dat přímo z portálu `openstreetmap.org` a poté stažení dat z portálu `https://extract.bbbike.org`.

OpenStreetMap

Po otevření portálu `openstreetmap.org` v levém horním rohu otevřeme klikem na tlačítko "Export"záložku pro export dat. Po stisku tlačítka export v levé části obrazovky je stažen soubor ve formátu *.osm*. Kliknutím na tlačítko "Ručne vybrat jinou oblast"může uživatel nastavit hranice pro výběr jiné oblasti. Pokud je vybrána oblast s více jak 50 000 body, otevře se stránka s chybovou hláškou "*You requested too many nodes (limit is 50000). Either request a smaller area, or use planet.osm*"

TODO: obrázky

BBBike

BBBike je portál umožňující stahovat větší oblasti, než je možné stáhnout z portálu `openstreetmap.org`. Navíc také umožňuje vybrat oblast polygonem, narozdíl od osm, kde lze vybrat oblast pouze tvaru obdelníku. Při stažení z portálu `bbbike.org` je nejprve nutné nastavit formát, ve kterém oblast bude stažena. K tomu je určeno menu v levém horním rohu. Ke správnému fungování je potřeba vybrat první formát "*Protocolbuffer (PBF)*". Poté je potřeba oblast pojmenovat. Jméno se vpíše do formuláře pod výběrem formátu. Na jméně oblasti nezáleží a uživatel si jej může zvolit libovolně. V další kolonce poté uživatel musí zadat e-mailovou adresu, na kterou přijde odkaz a pokyny ke stažení souboru.

TODO: obrázky

7.2 Spuštění OSM2TraMod

Ke spuštění musí mít uživatel nainstalovaný Docker. Stažení nástroje s pokyny k instalaci jsou dostupné na odkazu `docker.com/get-started/`.

Samotné spuštění probíhá pomocí shellscriptu *osm2tramod.sh*. Script se spouští z příkazové řádky. Uživatel musí zadat 5 parametrů. Těmi jsou:

- *AREA_NAME* - název oblasti
- *FORMAT* - formát vstupního souboru pbf/osm
- *IN_FILE* - název vstupního souboru
- *TILE_COUNT* - požadovaný počet dlaždic na rozdělení oblasti
- *OUT_DIR* - název výstupní složky

Spuštění tedy vypadá následovně:

```
./osm2tramod.sh \
        <AREA_NAME> \
        <FORMAT> \
        <IN_FILE> \
        <TILE_COUNT> \
        <OUT_DIR>
```

Příklad konkrétního spuštění může vypadat takto:

```
./osm2tramod.sh \
        Pilsen_vanek \
        pbf \
        IN_FILE/pilsen.osm.pbf \
        1000 \
        pilsen_output
```

Po skončení konverze je možné na cestě ve složce definované uživatelem možné nalézt pět sql souborů, které je možné použít pro import dat do nástroje TraMod. Tyto soubory jsou na uložené na cestě *<OUTPUT_FOLDER>/<AREA_NAME>/Tramod_data*. Jejich názvy jsou:

- *<AREA_NAME>_Tramod_Scheme.sql* - soubor pro vytvoření struktury databáze
- *<AREA_NAME>_Tramod_Edges.sql* - soubor s daty hran
- *<AREA_NAME>_Tramod_Vertexes.sql* - soubor s daty uzlů

- *<AREA_NAME>_Tramod_Turn_Restrictions.sql* - soubor s daty zakázaných směrů
- *<AREA_NAME>_Tramod_Zones.sql* - soubor s generátory dopravy

7.3 Import dat do nástroje TraMod

K importu dat do nástroje TraMod uživatel využívá libovolného SW pro práci s databázemi, který umožňuje práci s databází pomocí SQL souborů. Může být použit software ArcGIS, nebo například software DBeaver. K samotnému nahrání postačí i konzolový klient databáze PostgreSQL. Tento klient je volně dostupný na adrese `https://www.postgresql.org/download/`

Import do databáze pomocí konzolového klienta poté probíhá pomocí jednoduchého příkazu:

```
psql –h hostname –p port –d databasename –U username \
–f SQL_script_file_name}
```

Jednotlivé soubory musí být nahrávány ve stejném pořadí, jaké je uvedeno v následujícím seznamu:

1. *Scheme.sql* - soubor pro vytvoření struktury databáze
2. *Edges.sql* - soubor s daty hran
3. *Vertexes.sql* - soubor s daty uzlů
4. *Turn_Restrictions.sql* - soubor s daty zakázaných směrů
5. *Zones.sql* - soubor s generátory dopravy

Při pokusu o import v jiném pořadí může nastat situace, kdy nahrávané záznamy obsahují referenci na záznamy, které nejsou v databázi nahrány.

8 Dosažené výsledky

Implementované řešení je testováno na datasetu Plzeňského kraje a jeho okolí a na datasetu území Krumlovska. Proces jednotlivých transformací odpovídá kapitole 5. V této kapitole jsou u jednotlivých datasetů vypsány počty záznamů, které nástroj načítá a ve svém průběhu generuje. Na obrázcích jsou data v jednotlivých fázích zobrazena a jsou vybrána místa, na kterých je dokázána správná funkcionalita vlastně implementovaného nástroje.

Plzeňský kraj

Počet původních záznamů v souboru s daty OSM získaného z portálu `bbbike.org`:

- Vrcholy - 8 277 860
- Cesty - 756 646
- Relace - 19 730

Nástroj OSM2Po vlastní filtrací nastavené v konfiguraci vybere záznamy týkající se silniční sítě. Z původních dat tak extrahuje 402 894 vrcholů, 62 557 cest a 521 relací. Z extrahovaných dat poté vytvoří následující počet záznamů, reprezentující směrovatelnou silniční síť.

- Uzly (křižovatky) - 89 994
- Hrany - 107 961

K vytvoření generátorů jsou využity záznamy budov. Filtrace probíhá nástrojem osmfilter. Původní data OSM jsou filtrována a je tak extrahován následující počet záznamů reprezentující budovy:

- Vrcholy - 2 356 698
- budovy - 428 904

Vlastní nástroj během po zpracování dat vytvoří následující počet záznamů:

- Uzly - 89 994

- Hrany - 209 975
- Zakázané směry - 710
- Generátory dopravy - 681

Z 428 904 budov aplikace nepřiřadí 272 budov. Ztracení těchto dat je dáno jiným způsobem zápisu v původním souboru OSM. Některé budovy jsou v OSM zapsány pomocí relací, které referencují cesty. Při načítání cest tedy některé nemají počáteční a konečný uzel stejný a jsou pro výpočet zahozeny. Je tedy zahozeno 0,063% budov. Po bližším zkoumání zahozených dat bylo zjištěno, že většina z nich reprezentuje zastřešené autobusové zastávky se zanedbatelnou plochou. Takto nízká ztráta nemá na model zásadní vliv a je tedy ignorována.

V případě, že uživatel požaduje dělení oblasti na 1000 částí, aplikace vytvoří 681 generátorů. Chybějících 319 generátorů nebylo vytvořeno kvůli vymezení oblasti polygonem. Ve vytvořené mřížce, která má obdelníkový tvar tedy existovaly dlaždice, které na své vymezené ploše neobsahovaly žádná data, které by jim mohla být přiřazena. Chybějící generátory vně oblasti neexistují z důvodu neexistence budov na území, které pokrývají.

Takto generovaná data jsou nahrána do databáze nástroje TraMod.

Krumlovsko

Počet původních záznamů v souboru s daty OSM získaného z portálu `bbbike.org`:

- Vrcholy - 1 318 491
- Cesty - 116 193
- Relace - 3 335

Nástroj OSM2Po z původních dat tak extrahuje 61 463 vrcholů, 10 193 cest a 206 relací. Z extrahovaných dat vytvoří následující počet záznamů, reprezentující směrovatelnou silniční síť.

- Uzly (křižovatky) - 14 092
- Hrany - 16 286

Počet extrahovaných záznamů reprezentující budovy:

- Vrcholy - 337 084

- Budovy - 58 384

Vlastní nástroj během po zpracování dat vytvoří následující počet záznamů:

- Uzly - 14 092
- Hrany - 31 732
- Zakázané směry - 243
- Generátory dopravy - 262

Z 58 384 budov aplikace nepřiřadí 83 budov. Je tedy zahozeno 0,142% budov.

V případě, že uživatel požaduje dělení oblasti na 350 částí, aplikace vytvoří 262 generátorů. Chybějících 138 generátorů nebylo vytvořeno ze stejného důvodu jako u datasetu Plzeňského kraje.

8.1 Důkaz funkcionality

V této části jsou zobrazena data silniční sítě dokazující funkcionalitu jednotlivých transformací kapitoly 5.2.3 před a po konverzi vlastním nástrojem.

Zakázané směry

Na obrázku 8.1 je vidět křižovatka mezi ulicemi Ukrajinská, Nádražní, Tyršova a U Prazdroje. Tato křižovatka obsahuje velké množství zakázaných směrů a je tedy vybrána pro dokázáni funkcionality tvorby zakázaných směrů.

Po konverzi nástrojem OSM2Po získáme data znázorněna na obrázku 8.2. Zelené linie značí hrany a černé body značí uzly. Identifikátor hrany je vepsán přímo v linii tmavší zelenou barvou, identifikátor uzlu je popsán v jeho blízkosti černou barvou.

K ověření je využit uzel s identifikátorem 18, který po konverzi nástrojem OSM2Po obsahuje v atributu *restrictions* obsahuje následující hodnotu:

−69638_12−12_12−69638_69627−69625_69627−69625_69626
−12_69626

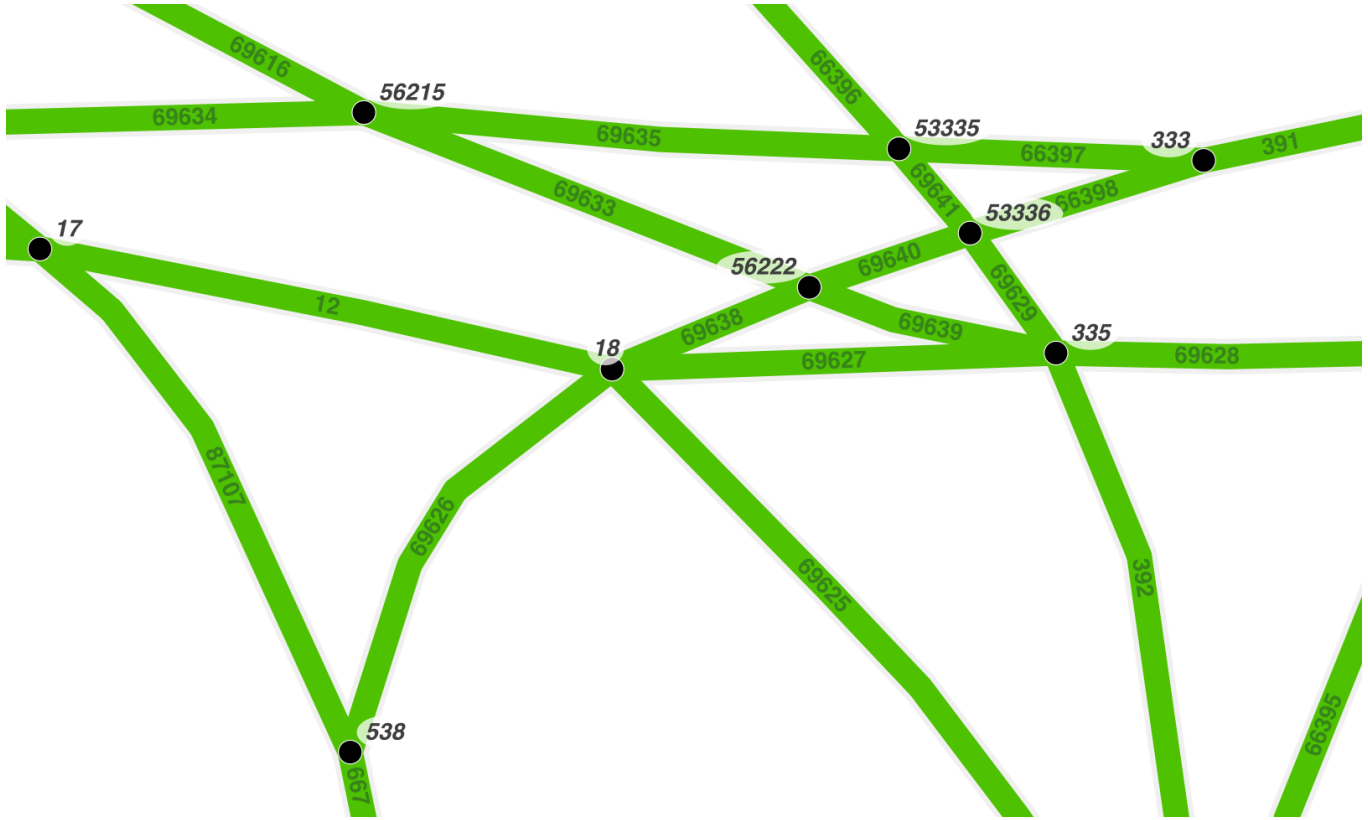
Jsou tedy zakázány směry v tabulce 8.1

Vlastní konverzní nástroj vytvoří záznamy se zakázanými směry procesem popsaným v kapitole 5.2.3. Finální data jsou graficky znázorněna na obrázku 8.3. Hrany jsou znázorněny oranžovou linií a jejich identifikátor je po pravé straně směru hrany současně s vyznačeným směrem.

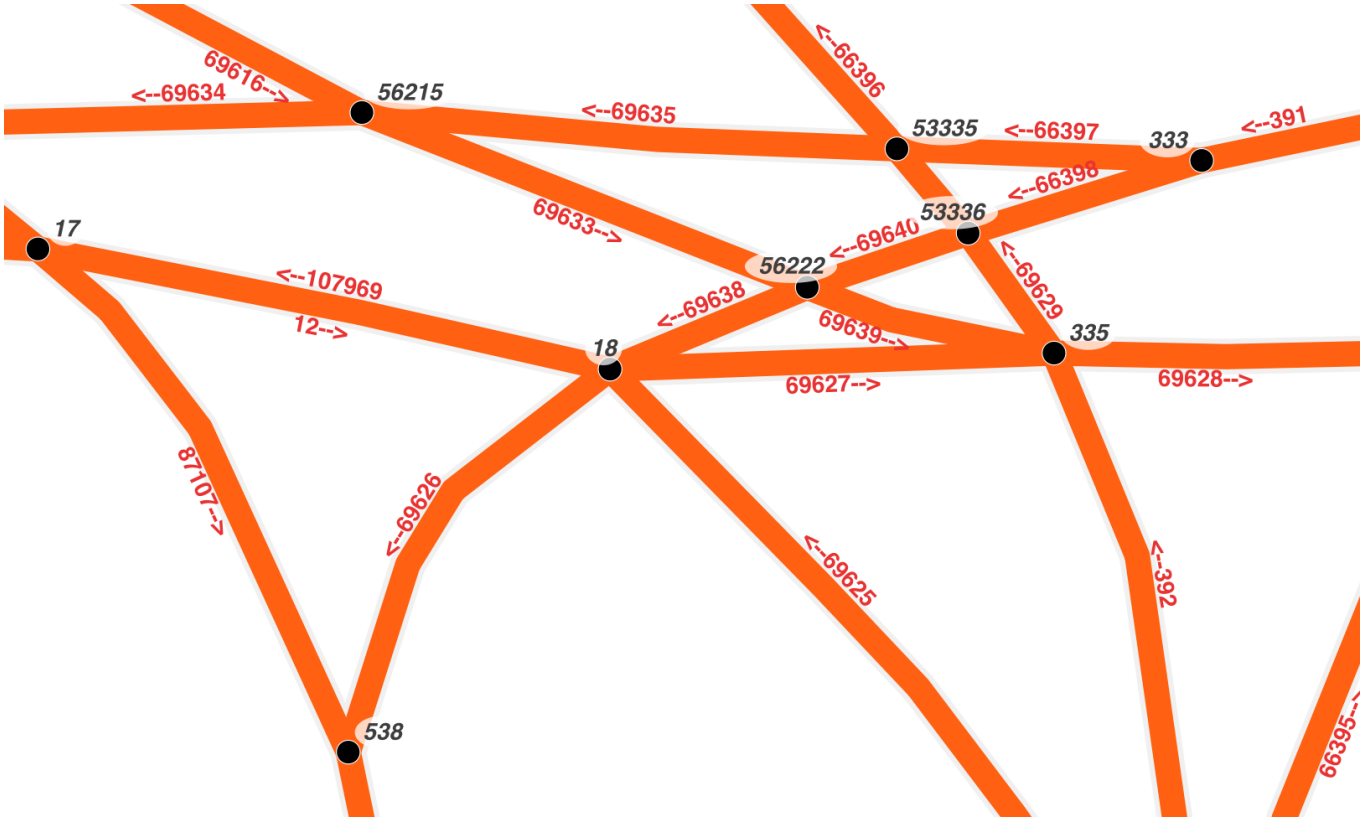
Obrázek 8.1: vybraná křižovatka

Zakázané směry - OSM2Po	
source id	target id
69638	12
12	12
69638	69627
69625	69626
12	69626

Tabulka 8.1: tabulka zakázaných směrů



Obrázek 8.2: Data po konverzi vlastním nástrojem



Obrázek 8.3: Data po konverzi vlastním nástrojem

9 Diskuze

Tato kapitola se zabývá nedostatky jednotlivých datových struktur a postupů, které byly během práce objeveny. Jsou zde uvedeny jednotlivé problémy, jejich následky a navrhované řešení.

Export dat OSM2Po

Nástroj OSM2Po slouží k vytvoření směrovatelné silniční sítě. Tento nástroj po svém spuštění dokáže exportovat data pouze jako SQL Dump soubory obsahující příkazy v jazyce SQL, které se dají použít pro nahrání dat do databáze. Při nahrávání dat ze souboru vlastním nástrojem je tedy nutné využít vlastní parser, který data ze souboru načte. Pokud by nástroj OSM2Po umožňoval export dat do souboru v jiném formátu (XML, JSON), načtení dat by se značně zjednodušilo.

API nástroje Traffic Modeller

Nástroj Traffic Modeller umožňuje práci s uloženými daty přes REST API. API umožňuje získávat data z nástroje Traffic Modeller. Jeho pomocí je také možné nahrávat jednotlivé záznamy do datové struktury. API však neumožňuje vytvořit nový model. Pokud tedy vytvoříme data pro území, které v nástroji Traffic Modeller není uložené v již existujícím modelu, nelze je do nástroje bez vytvoření modelu nahrát. Data tedy musí být uložena do databáze a o migraci dat do nástroje Traffic Modeller se starají administrátoři nástroje.

API nástroje umožňuje rozsáhlou práci s jeho daty. Přidání funkcionality pro vytvoření nového modelu by tak umožnilo nahrání dat přímo do nástroje.

Hodnota generátorů dopravy

Jak bylo zmíněno v kapitole 5.2.3 odhad hodnoty generátoru dopravy zavisí na různých demografických faktorech a lokálních parametrech. Tato práce pro odhad využívá pouze plochu jednotlivých budov v m^2. Tento odhad by se měl měnit na základě různých typů budov, počtu jejich pater a dále. Algoritmus pro tento výpočet není definován a není tedy implementován. Vypočtená hodnota tedy nekoresponduje s reálným odhadem.

Po dodání algoritmu je možné změnit implementaci výpočtu generátorů na základě atributů budov, což zpřesní odhad generované dopravy.

Oblasti generátorů

Generátory dopravy jsou počítány pro jednotlivé oblasti. Pro účely diplomové práce bylo území rozděleno na čtvercovou mřížku, kde každá dlaždice reprezentuje jednu oblast. Jednotlivé dlaždice tak ohraničují stejně velké oblasti ve městech s velkým počtem budov i mimo města, kde se téměř žádné budovy nevyskytují. Problémy mohou vznikat zvlášť v místech, kde skrz oblast vede například řeka, která oblast rozděluje. Může se tedy stát, že nástrojem je vypočítán odhad generátorů na jedné straně řeky a je přiřazen křižovatce na straně druhé.

Nevhodné je také přiřazení generátorů křižovatce nejblíže středu dlaždice. Generátory by měly být přiřazeny křižovatkám s vyšším tokem dopravy, aby lépe reprezentovaly reálnou situaci. Může se tedy stát, že část území uvnitř dlaždice může generovat vysokou hustotu dopravy, ale generátor může být přiřazen nějaké nevýznamné křižovatce mimo toto území. Toto přiřazení také úzce souvisí právě s rozdělením území na oblasti.

Algoritmus pro vhodnější rozdělení oblasti také není definován. Vhodnějším způsobem rozdělení území na menší oblasti mohou být Voroného diagramy. Po zvolení či definování vhodného algoritmu pro rozdělení oblastí může být algoritmus implementován, což zpřesní odhad generované dopravy.

Návrh zpětného algoritmu

Jak bylo popsáno v kapitole 5.3, nástroj Traffic Modeller nijak neposkytuje informaci o intenzitě dopravy. Data OSM o tuto informaci tedy v současné době nelze obohatit. REST API nástroje umožňuje získat data o jednotlivých hranách. Pokud do těchto dat byla přidána i hodnota intenzity dopravy, umožnilo by to obohacení původních dat OSM. Možnost tuto hodnotu skrz API získat je tedy stěžejní k implementaci navrhovaného algoritmu.

10 Závěr

V diplomové práci byla obecně popsána geografická data a ETL proces na ně zaměřený. Byly představeny a detailně popsány datové struktury OpenStreetMap a nástroje Traffic Modeller. Byl navržen a implementován nástroj schopný konverze dat z datové struktury OpenStreetMap do datové struktury nástroje Traffic Modeller. Byl popsán proces konverze mezi datovými strukturami využívaný implementovaným nástrojem, včetně analýzy a popisu externích modulů, které byly pro konverzi použity. Konverzní nástroj byl testován na dvou pilotních územích. Těmito testy byla prokázána správná funkcionalita. Výsledné datasety byly nahrány do požadované databáze nástroje Traffic Modeller. Během testů byla zjištěna ztráta méně než 0,2% dat, která nemá zásadní vliv na kvalitu výsledného modelu.

Výsledný nástroj byl dockerizován, což umožňuje jeho využití na serveru. Součástí diplomové práce byla vytvořena uživatelská dokumentace popisující způsob použití implementovaného konverzního nástroje s popisem povinných i volitelných vstupů.

Vytvořený nástroj dokáže převést data OpenStreetMap libovolného území do datové struktury vyžadované nástrojem Traffic Modeller. Toto umožňuje automatizace přípravy dat pro simulaci různých dopravních situací na libovolném území nástrojem Traffic Modeller. Tato práce dokázala, že odhad intenzity dopravy lze ze zdrojových dat uskutečnit, avšak kvůli nedostatkům popsaných v diskuzi je nutná kalibrace odhadu pro zvýšení jeho kvality. Kvalitnějšího modelu lze dosáhnout definicí a implementací přesnějších algoritmů pro výpočet jednotlivých generátorů, rozdělení oblasti a přiřazení generátorů křižovatkám do vytvořeného konverzního nástroje.

Literatura