

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA STAVEBNÍ
PROGRAM GEODÉZIE A KARTOGRAFIE
OBOR GEOMATIKA



DIPLOMOVÁ PRÁCE
ROZŠÍŘENÍ ZÁSUVNÉHO MODULU QGIS
PRO ZPRACOVÁNÍ DAT GTFS O VIZUALIZACI TARIFNÍCH
PÁSEM PID

Vedoucí práce: Ing. Martin Landa, Ph.D.
Katedra geomatiky

červen 2021

Bc. Martin KOUBA

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: <u>Kouba</u>	Jméno: <u>Martin</u>	Osobní číslo: <u>468232</u>
Zadávatel katedra: <u>Katedra geomatiky</u>		
Studijní program: <u>Geodézie a kartografie</u>		
Studijní obor: <u>Geomatika</u>		

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce: <u>Rozšíření zásuvného modulu QGIS pro zpracování dat GTFS o vizualizaci tarifních pásem PID</u>	
Název diplomové práce anglicky: <u>QGIS GTFS Plugin extension with visualization of PID tariff zones</u>	
Pokyny pro vypracování: Cílem diplomové práce je rozšířit zásuvný modul (plugin) platformy QGIS pro zpracování dat ve formátu GTFS, na jehož vývoji se autor spolupodílel v rámci semestrálního projektu předmětu Free Software GIS ve 2. semestru magisterského studia. Primárním úkolem je najít a vhodně implementovat způsob vizualizace tarifních pásem Pražské integrované dopravy (PID) generovaných automaticky z podkladových dat.	
Seznam doporučené literatury: SHERMAN, Gary. The PyQGIS Programmer's Guide: Extending QGIS 3 with Python 3. Ilustrované vydání. Locate Press, 2018. ISBN 9780998547725. MARK, Pilgrim. Dive into Python 3. Apress, 2009. ISBN 978-1430224150.	
Jméno vedoucího diplomové práce: <u>Ing. Martin Landa, Ph.D.</u>	
Datum zadání diplomové práce: <u>10.2.2021</u>	Termín odevzdání diplomové práce: <u>17.5.2021</u> <i>Údaj uveďte v souladu s datem v časovém plánu příslušného ak. roku</i>
_____	_____
Podpis vedoucího práce	Podpis vedoucího katedry

III. PŘEVZETÍ ZADÁNÍ

<i>Beru na vědomí, že jsem povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je nutné uvést v diplomové práci a při citování postupovat v souladu s metodickou příručkou ČVUT „Jak psát vysokoškolské závěrečné práce“ a metodickým pokynem ČVUT „O dodržování etických principů při přípravě vysokoškolských závěrečných prací“.</i>	
_____	_____
Datum převzetí zadání	Podpis studenta(ky)

ABSTRAKT

Tato diplomová práce se zabývá automatizací vytváření tarifních pásem pro dopravní systém Pražské integrované dopravy. Implementace řešení je prováděna jako rozšíření již existujícího zásuvného modulu za pomoci GTFS datasetu. Automatizace by měla pomoci zaměstnancům organizace ROPID s tvorbou tarifních pásem, která se provádí téměř celá manuálně. Rozšíření zásuvného modulu bylo vytvořeno pomocí programovacího jazyka Python v open source systému QGIS.

KLÍČOVÁ SLOVA

QGIS, zásuvný modul, GTFS, PID, tarifní pásma, Python

ABSTRACT

This Master's thesis deals with the automation the creating of tarrif zones for a public transport system of Prague Integrated Transport. The implementation of the solution is performed by an extension of an existing plugin using the GTFS dataset. This automation should help ROPID staff with the creation of the tariff zones. The creation is carried out almost entirely on a manual basis. This plugin extension was created using the Python programming language in the open-source QGIS system.

KEYWORDS

QGIS, plugin, GTFS, PID, tariff zones, Python

PROHLÁŠENÍ

Prohlašuji, že diplomovou práci na téma „Rozšíření zásuvného modulu QGIS pro zpracování dat GTFS o vizualizaci tarifních pásem PID“ jsem vypracoval samostatně. Použitou literaturu a podkladové materiály uvádím v seznamu zdrojů.

V Praze dne

.....

(podpis autora)

PODĚKOVÁNÍ

Tímto bych chtěl poděkovat panu Ing. Martinu Landovi, Ph.D. za veškerou pomoc a strávený čas při tvorbě diplomové práce. Dále chci poděkovat doc. Ing. Tomáši Bayerovi, Ph.D. za odborné rady k algoritmům digitální kartografie, Ing. Alžbetě Gardoňové za konstruktivní nápady a pomoc s komunikací se zaměstnanci organizace ROPID Ing. Zuzaně Šaškové a RNDr. Zbyňkovi Jiráčkovi, kterým chci také poděkovat za podávání pravidel a informací o tarifních pásmech Pražské integrované dopravy. Zvláště bych chtěl poděkovat celé své rodině a přítelkyni za trpělivost a pomoc při studiu.

Obsah

Úvod	11
1 Rešerše	12
2 GTFS	21
2.1 Historie GTFS	21
2.2 GTFS dataset	22
2.2.1 Soubor stops.txt	24
3 Pražská Integrovaná doprava (PID)	27
3.1 ROPID	27
3.2 Tarifní pásma PID	28
3.2.1 Tvorba tarifních pásem PID	29
4 Použité technologie	31
4.1 QGIS	31
4.2 Python	32
4.3 PyQt	33
4.4 Ostatní	33
4.4.1 GitHub	33
4.4.2 PyCharm	34
5 Praktická část	35
5.1 Aktualizace pluginu do verze 1.0	35
5.2 Tvorba vizualizace tarifních pásem (experimentální verze pluginu 2.0)	35
5.2.1 Dialogové okno pro tvorbu tarifních pásem	36
5.2.2 Hrubý tvar tarifních pásem	37
5.2.3 Vyhlazení tvaru tarifních pásem	42
5.2.4 Hraniční tarifní pásma	47
5.2.5 Složení tarifních pásem	50
5.2.6 Symbologie tarifních pásem	51
Závěr	53

Seznam zkratk	55
Odkazy	56
A Dodatek	59
A.1 Instalace pluginu	59
A.1.1 Instalace pluginu pomocí QGIS repozitáře	59
A.1.2 Instalace pluginu pomocí ZIP souboru	59
A.2 Použití pluginu pro vytvoření tarifních pásem	60

Seznam obrázků

1.1	Ilustrace Greedy triangulace	12
1.2	Ilustrace Delaunay triangulace	13
1.3	Konvexní obálka	14
1.4	Konvzávní obálka pomocí alpha-shapes	15
1.5	Terminologie Voronoi diagramu	16
1.6	Voronoi polygony	17
1.7	Znázornění algoritmu „Douglas-Peucker“	19
1.8	Znázornění Chaikinova algoritmu	20
2.1	Ukázka CSV formátu ze souboru shapes.txt	22
2.2	Diagram GTFS datasetu	24
2.3	Ukázka CSV souboru stops.txt pro PID_GTFS dataset	26
3.1	Logo Pražské integrované dopravy	27
3.2	Logo organizace ROPID	27
3.3	Schéma tarifních pásem PID	28
3.4	Nejstarší dostupná verze tarifních pásem PID	30
3.5	Tarifní pásma zveřejněná organizací ROPID z 8. září 2020	30
4.1	Logo systému QGIS	31
4.2	Logo Pythonu	32
4.3	Logo GitHubu	33
4.4	Logo PyCharmu	34
5.1	Logo pluginu GTFS Loader	35
5.2	Ukazatel průběhu (ProgressBar) v liště systému QGIS	35
5.3	Dialogové okno s možností tvořit/netvořit tarifní pásma	36
5.4	Voronoi polygony pro všechny zastávky	37
5.5	Vybrané Voronoi polygony pro tarifní pásma P, 0 a B	38
5.6	Výsledek nástroje Dissolve pro tarifní pásma P, 0 a B	39
5.7	Výsledek nástroje Delete holes pro tarifní pásma P, 0 a B	40
5.8	Postup pro tarifní pásma P, 0 a B	41
5.9	Postup pro tarifní pásma 1 až 9	42
5.10	Výsledek nástroje Extract vertices	43

5.11	Spojené body třech vektorových vrstev pomocí nástroje Merge vector layers pro tarifní pásma P, 0 a B	44
5.12	Konkávní obálka	45
5.13	Výsledek nástroje Simplify pro tarifní pásma P, 0 a B	46
5.14	Výsledek nástroje Smooth pro tarifní pásma P, 0 a B	46
5.15	Postup pro zjemnění tvaru tarifních pásem pro tarifní pásma	47
5.16	Ukázka upraveného nástroje Concave hull (alpha shapes) pro část tarifních pásem P, 0 a B	48
5.17	Postup pro hraniční tarifní pásma	50
5.18	Postup pro složení vektorových vrstev tarifních pásem	51
5.19	Výsledný tvar tarifních pásem	51
A.1	Dialogové okno pro výběr instalace pluginu z QGIS repozitáře	59
A.2	Dialogové okno pro výběr ZIP souboru	60
A.3	Dialogové okno pluginu GTFS Loader	60
A.4	Výsledek procesu pluginu	61

Seznam tabulek

2.1	Seznam CSV souborů v GTFS datasetu	23
-----	--	----

Úvod

Pro cestování po Praze a okolí je často nejlepší volbou použít veřejnou dopravu. Tu zahrnuje metro, tramvaje, železnici, městské a příměstské autobusové linky. Kromě těchto dopravních prostředků je do veřejné dopravy zahrnuta lanová dráha na Petřín a také některé přívozy. Celý systém se nazývá Pražská integrovaná doprava. Postupně je integrován společnými přepravními a tarifními podmínkami a jednotným dopravním řešením včetně koordinace jízdních řádů.

Systém Pražské integrované dopravy je rozdělen do dvanácti tarifních pásem, pro které platí různé jízdní doklady. Rozlohou jsou po celé Praze, na většině Středočeského kraje a dokonce i na částech Plzeňského a Ústeckého kraje, či na Vysočině. Více o systému Pražské integrované dopravy je uvedeno v kapitole 3.

Postup tvorby tarifních pásem Pražské integrované dopravy je poloautomatizovaný a manuální zásah tvůrce je tak téměř ke všemu nutný. Primárním cílem této diplomové práce bylo za pomoci zastávek z GTFS datasetu vytvořit postup a jeho následnou implementaci pro automatické vyhotovení exaktního modelu tarifních pásem. Implementace postupu se měla vkládat do předpřipraveného zásuvného modulu v systému QGIS. Výsledek procesu tvorby se měl co nejvíce přiblížit k oficiální podobě tarifních pásem vydávanou organizací ROPID. Oficiální podoba tarifních pásem je veřejně publikována na portálu *Opendata hlavního města Prahy*. O GTFS datasetu se píše v kapitole 2.

Nutně vyhotovitelným mezikrokem k dosažení primárního cíle bylo dokončení vývoje zásuvného modulu do uživatelsky přívětivější verze, která mohla být nahrána do QGIS repozitáře. Vývoj zásuvného modulu započal s mými spolužáky Adrianou Brezničanovou a Jaroslavem Zemanem v 5. semestru magisterského studia v předmětu Free software GIS. O softwaru QGIS a dalších použitých technologiích se pojednává v kapitole 4.

Sekundárním cílem bylo vytvořit schématický model, který nesplňuje tolik pravidel pro tvorbu tarifních pásem, ale má informační hodnotu pro mapy menších měřítek, například pro různé plánky Pražské integrované dopravy.

V poslední kapitole 5 je popsán proces tvorby rozšíření zásuvného modulu. Zahrnuje i několik nepodařených pokusů, které byly v rámci tvorby diplomové práce vyzkoušeny a následně zavrhnuty kvůli zjištění vhodnější verze postupu.

1 Rešerše

Pro přiblížení se k oficiálně publikovaným tarifním pásmům byly potřeba využít algoritmy digitální kartografie. Tarifní pásma jsou publikována mimo jiné jako shapefile ¹, který obsahuje vektorové geometrické data ve formě polygonových prvků. K této podobě bylo potřeba přistoupit za pomoci některých vybraných algoritmů.

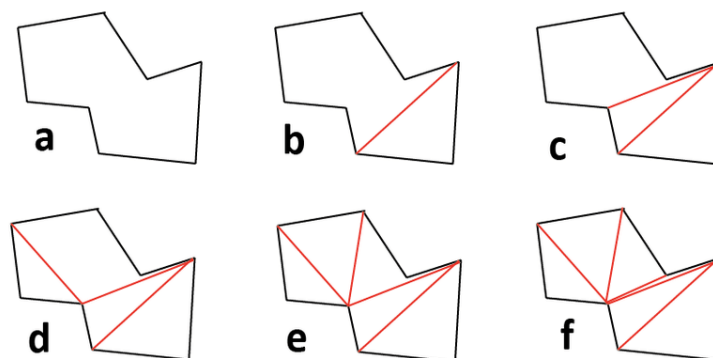
Triangulace

K vytvoření linií mezi jednotlivými zastávkami slouží takzvaná triangulace, která vytvoří trojúhelníky mezi body tak, že uvnitř těchto trojúhelníků už neleží žádné body a každý trojúhelník má vždy společnou jednu hranu.

Triangulace se používají jak v kartografii, tak v GIS, v Dálkovém průzkumu země (DPZ), počítačové grafice, při analýze vlastností a struktury materiálů, plánování pohybu robotů nebo při modelování přírodních jevů. [2]

Existuje více druhů triangulací, které využívají vždy jinou metodu konstrukce a mají rozdílný výpočetní stupeň složitosti.

Hladová (Greedy) triangulace se snaží vytvářet trojúhelníky s nejkratšími stranami, které nesplňují žádnou speciální geometrickou podmínku. Její realizace je jednoduchá, avšak důsledkem toho jsou často tvarově nepěkné nebo nevhodné trojúhelníky. Má velkou výpočetní složitost $O(n^3)$, lze optimalizovat na $O(n^2 \log(n))$. V kartografii se kvůli její složitosti často nepoužívá. [3]

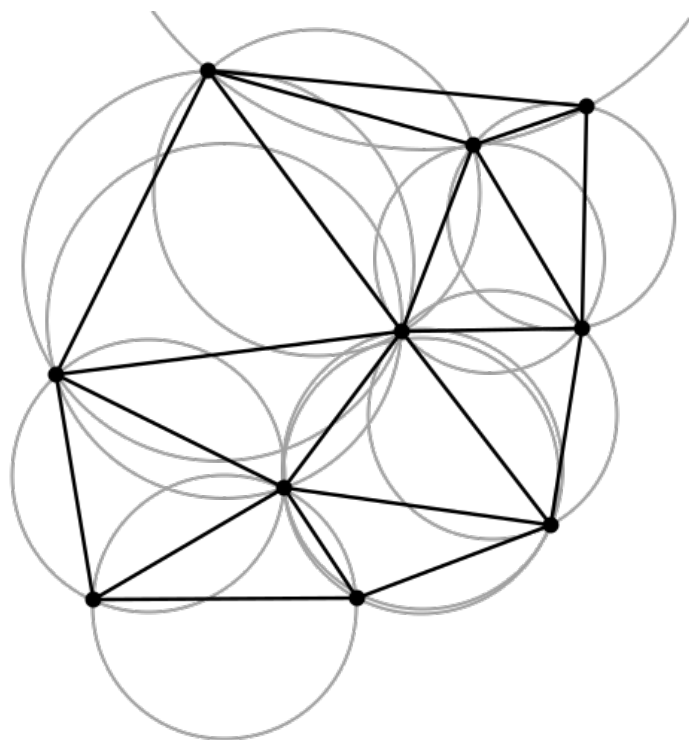


Obrázek 1.1: Ilustrace Greedy triangulace [4]

¹formát vektorového datového úložiště Esri pro ukládání umístění, tvaru a atributů geografických prvků [1]

Další triangulací je tzv. *Delaunay triangulace*, která je nejčastěji používaná. *Delaunay triangulací* se rozumí vytváření liniových spojníc bodů mezi jednotlivými body, které jsou si nejbližší, za pomoci opsaných kružnic. *Delaunay triangulace* má několik vlastností. Jednou z nich je například, že uvnitř kružnice k opsané libovolnému trojúhelníku neleží žádný jiný bod. Zároveň *Delaunay triangulace* je jednoznačná, pokud žádné čtyři body neleží na kružnici. Na rozdíl od *Greedy triangulace* nehodnotí kritérium délky hran. Díky maximalizaci minimálních úhlů vytváří takové trojúhelníky, které se nejvíc blíží k rovnostranným trojúhelníkům, což znamená, že se snaží eliminovat trojúhelníky, které jsou ostroúhlé.

Pro vytváření konstrukce *Delaunay triangulace* jsou k dispozici různé algoritmy: lokální prohazování, inkrementální konstrukce, inkrementální vkládání, rozděl a panuj nebo sweep line. [2]



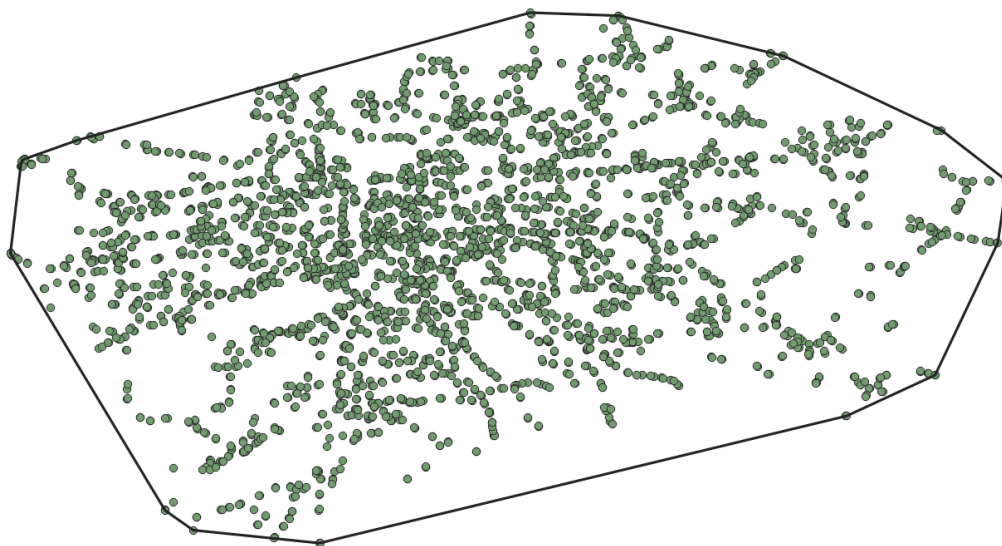
Obrázek 1.2: Ilustrace Delaunay triangulace [5]

Poté existuje *triangulace s minimální hmotností* (anglicky *Minimum Weight Triangulation*, zkráceně MWT), která má minimální celkovou délku hran. MWT je nejlepší triangulace pro interpolaci trojrozměrných oblastí, tudíž se pro tvorbu 2D tarifních pásem úplně nehodí.

Konvexní obálka

Pro vytváření hranic polygonů je konvexní obálka jedna z možností, co použít. Konvexní obálka (anglicky Convex hull) je nejmenší konvexní množinou, pokud spojnice libovolných dvou prvků leží zcela uvnitř této množiny. Je to jedna z nejpoužívanějších geometrických struktur, která je většinou používána jako první odhad prostorového tvaru.

Konstrukce konvexní obálky se nejčastěji provádějí metodami: Jarvis Scan, Graham Scan, QuickHull nebo rozděl a panuj. Konvexní obálku lze vytvářet i pomocí *Delaunay triangulace*, kdy se po jejím dokončení spojí všechny trojúhelníky.



Obrázek 1.3: Konvexní obálka

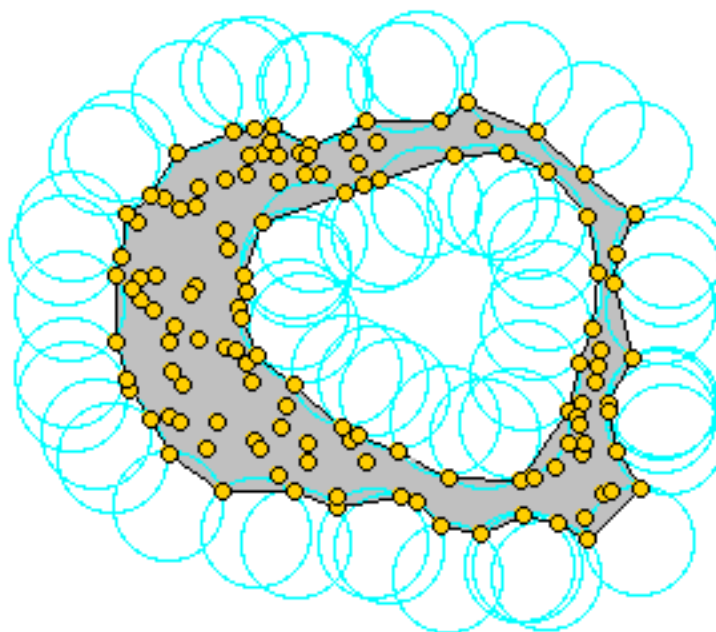
Konkávni obálka

Oproti konvexní obálce je konkávni obálka pro tvorbu složitější. Není znám přesný postup, jak tento problém řešit, ale existují různé aproximace, které se k tomu chtějí přiblížit. První aproximace je pomocí *Delaunay triangulace*, kde se vytvoří trojúhelníky, vznikne konvexní obálka a pak se odebírají trojúhelníky s nevhodnými vlastnostmi. Typicky mají příliš ostrý nebo tupý úhel, hraniční hodnota vlastností se může zvolit. Předpokladem pro úspěšně fungování je to, že množina bodů má konstantní prostorovou hustotu bodů.

Další aproximací je pomocí *alpha-shapes*, kde se volí parametr α . Představme si obrovskou masu zmrzliny, která tvoří prostor R^3 a obsahující body jako „tvrdé“ kousky čokolády. Pomocí jedné z těchto zmrzlinových lžiček ve tvaru koule

vyřezáme všechny části zmrzlinového bloku, na které se dostaneme, aniž bychom narazili na kousky čokolády, a tím dokonce vydlabeme otvory uvnitř (např. části nedosažitelné pouhým pohybem lžice z venku). Nakonec skončíme s (ne nutně konvexním) objektem ohraničeným oblouky a body. Pokud nyní narovnáme všechny „kulaté“ plochy na trojúhelníky a úsečky, máme intuitivní popis toho, co se nazývá alpha-shape. [6]

Další strategií je kompromisní definice mezi alpha-shapes a konkávní obálkou, tzv. alpha-concave hull.



Obrázek 1.4: Konkávní obálka pomocí alpha-shapes [7]

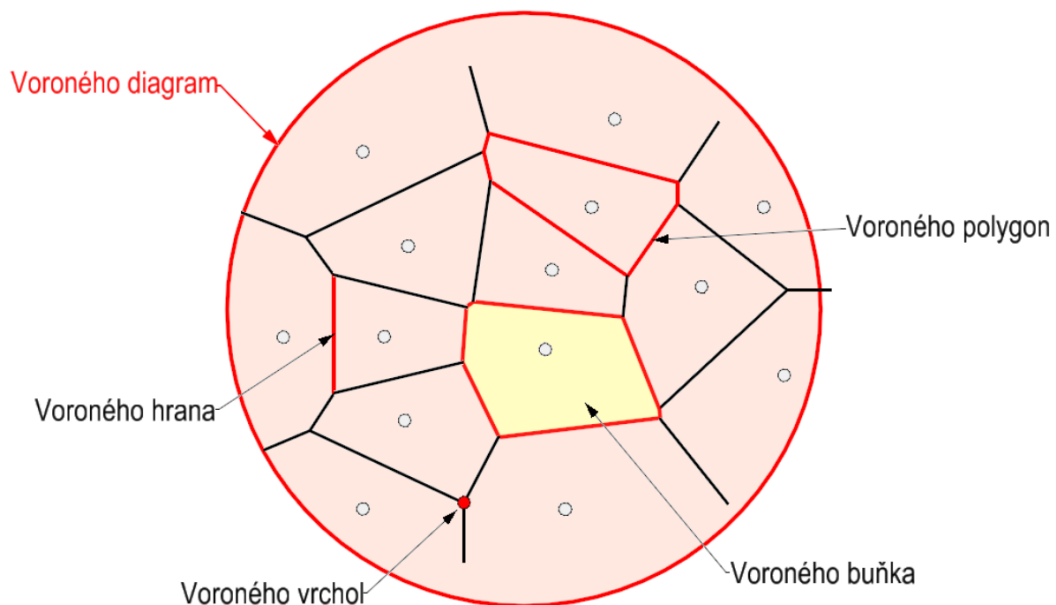
Voronoi diagram

Tarifní pásma zasahují hranicí tam, kde nejsou žádné zastávky. Pro takové vyplnění plochy mezi zastávkami a pro místa, kde zastávky nejsou (většinou na krajích zón) není triangulace, konvexní nebo konkávní obálka úplně vhodná. Pro takový problém bylo další myšlenkou použít teselaci, což je vyplnění roviny pomocí jednoho nebo více geometrických útvarů vzájemně spojených, bez překryvů a mezer. Triangulace, konvexní nebo konkávní obálka se mohou hodit při dalších krocích výpočtu.

Nejpoužívanější teselací v oblastí GIS je *Voronoi diagram*, někdy nazývána Voroného teselace, Voroného dekompozice, Thiessenovy polygony nebo Dirichletova teselace, což je způsob rozkladu metrického prostoru určený vzdálenostmi k dané nespojitě množině bodů v prostoru. V našem případě se bude řešit Voronoi diagram ve 2D prostoru, tedy v rovině.

Na vstupu je nějaká množina bodů a výstupem je *Voronoi diagram*, což představuje takovou množinu buněk, pro které bude platit, že každý bod q náležící množině $V(p_i)$ je blíže k bodu p_i než k jakémukoliv bodu p_j náležící množině P . [8]

Pro lepší chápání *Voronoi diagramů* je potřeba si vysvětlit její terminologii. Vstupní množinu bodů nazýváme generátory, každý bod generuje jednu Voronoi buňku. V terminologii GIS se hovoří o tzv. Voronoi polygony. Tyto Voronoi buňky jsou tvořeny hranami, které spojují dva Voronoi vrcholy. Dohromady tyto Voronoi buňky tvoří Voronoi diagram.



Obrázek 1.5: Terminologie Voronoi diagramu [8]

Voronoi diagramy mají své vlastnosti, které jsou důležité pro jejich tvorbu. V následujících odrážkách jsou doslovně citována jejich znění z prezentace pana doc. Ing. Tomáše Bayera, Ph.D.

- Voronoi diagram $V(P)$ je planárním grafem.
- Vrchol q Voronoi buňky $V(p_i)$ je průnikem 3 hran, právě když je $V(P)$ nedegenerovaný.
- Pokud $p_i \in H(P)$, pak je $V(p_i)$ otevřený.

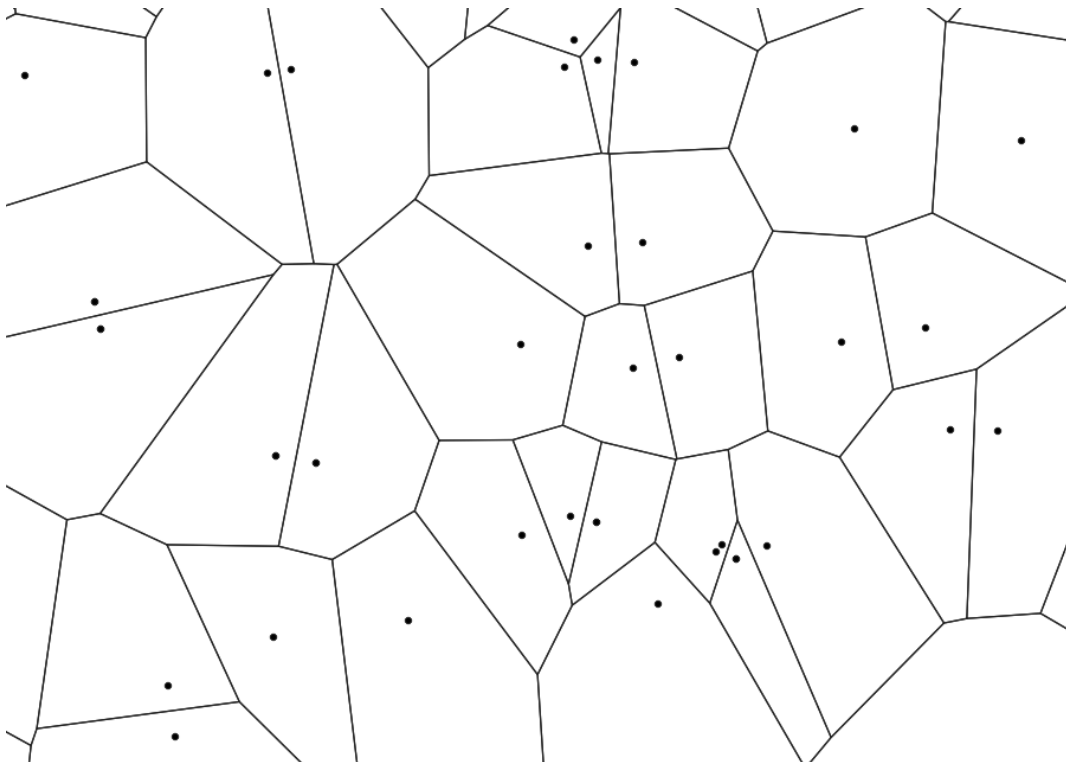
- Pro každý bod $p_i \in P$ je $V(P)$ konvexní.
- Bod p_i je nejbližším bodem bodu p jestliže $p \in V(p_i)$.
- Každá strana $q_i q_j$, $i \neq j$, je sdílena právě dvěma sousedními buňkami $V(p)$.
- Bod q je vrcholem $V(p)$, pokud existuje kružnice $k(q,r)$ procházející třemi nebo více generátory p_i, p_j, p_k , a neobsahuje žádný další bod P (spojitost s $DT(P)$).
- Kružnici $k(q,r)$ označujeme jako největší prázdnou kružnici ze všech prázdných kružnic se středem v bodě q .
- Průměrné množství Voronoi hran ve Voronoi polygonu nepřekročí hodnotu 6.
- Vztah mezi počtem bodů n , počtem hran n_h a počtem trojúhelníků n_t teselace $V(P)$:

$$n_h \leq 3n - 6$$

$$n_t \leq 2n - 5$$

- Voronoi diagram $V(P)$ představuje ortografickou projekci stěn mnohostěnu tvořeného průsečnicemi všech polorovin A_i do roviny xy .
- Nechť bod p_i^* představuje ortografický průmět bodu p_i na povrch paraboloidu daného rovnicí:

$$z = x^2 + y^2$$



Obrázek 1.6: Voronoi polygony

Voronoi diagram má spojitost s *Delaunay triangulací*. Hraniční vstupní body *Voronoi diagramu* po spojení *Delaunay triangulací* tvoří konvexní obálku. Střed

kružnic opsaných trojúhelníků *Delaunay triangulace* představují uzlové body *Voronoi diagramu*.

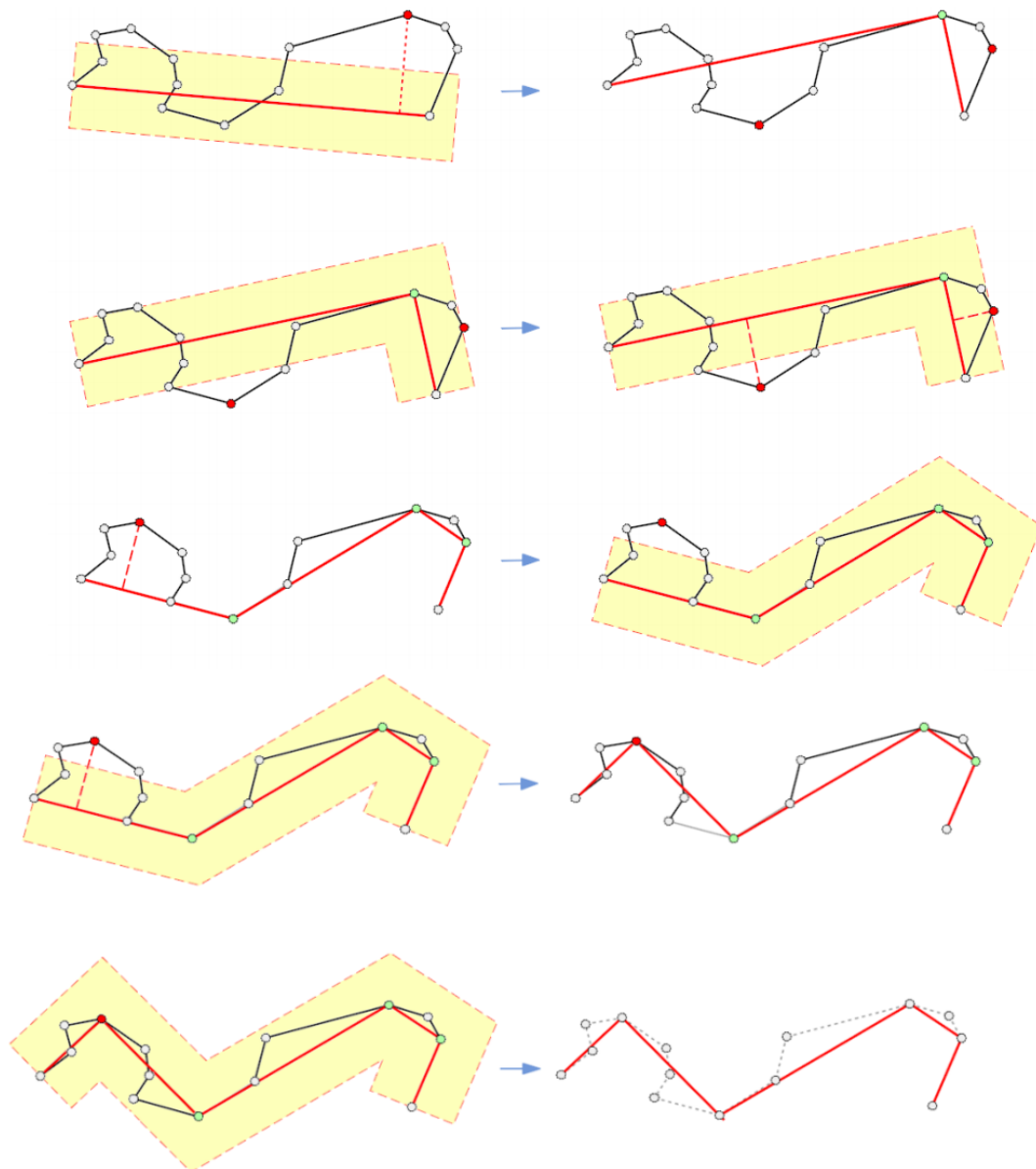
Pro konstrukci *Voronoi diagramu* existují různé metody. Konstrukce lze v praxi tvořit přímo nebo nepřímo. Pro přímou konstrukci jsou tu metody inkrementální konstrukce, sweep line algoritmus a rozděl a panuj. Nepřímá konstrukce je vytvářena přes *Delaunay triangulaci* skrz spojení středů kružnic opsaných trojúhelníků a bývá nejpoužívanější.

Využití Voronoi diagramů v oblasti GIS je široké. Například tzv. poštovní problém pomáhá při návrhu nových supermarketů, stanic MHD či polohy nových nemocnic. Mezi další využití patří například převod bodových údajů na plošné nebo klasifikace dat. [8]

Generalizace a zjemnění

Pro přiblížení se výsledku k graficky estetičtější verzi je potřeba hranice polygonů generalizovat. Následně se hranice zjemní. K tomu existuje mnoho algoritmů, ale bude zde uvedeno jen několik vhodných a nejpoužívanějších.

Jedním z nich je generalizační algoritmus *Douglas-Peucker*. Algoritmus rekurzivně rozděljuje čáru. Zpočátku jsou uvedeny všechny body mezi prvním a posledním bodem. Automaticky označí první a poslední bod, který se má zachovat. Poté najde bod, který je nejvzdálenější od úsečky s prvním a posledním bodem jako koncovými body. Tento bod je zjevně nejvzdálenější na křivce od přibližné úsečky mezi koncovými body. Pokud je bod blíže k úsečce, než volitelný parametr epsilon, mohou být všechny body, které nejsou aktuálně označeny jako zachované, zahozeny, aniž by rozměr zjednodušené křivky byl menší než parametr epsilon. Pokud je nejvzdálenější bod od úsečky větší než parametr epsilon od aproximace, musí být tento bod zachován. Algoritmus rekurzivně volá sebe samotného s prvním bodem a nejvzdálenějším bodem a poté nejvzdálenějším bodem a posledním bodem, který zahrnuje nejvzdálenější bod označený jako ponechaný. Tento algoritmus je jeden z nejlepších generalizačních algoritmů a je velmi často implementován v GIS softwarech. [9]



Obrázek 1.7: Znárodnění algoritmu „Douglas-Peucker“ [9]

Dalším algoritmem je zjemňovací/vyhlazovací algoritmus *Polynomická aproximace metodou Exponenciálního jádra* (anglicky *Polynomial Approximation with Exponential Kernel*, zkráceně PAEK). Tento algoritmus vyvinula společnost Esri pro nástroj Smooth Line v systému ArcGIS a je jejím soukromým vlastnictvím. Ačkoliv se zdá být ideální, tak není použitelný pro jiné platformy kvůli nezveřejněnému algoritmu.

Podobným algoritmem jako PAEK je algoritmus *Chaikinův*. Chaikin (autor algoritmu, po kterém je i algoritmus pojmenován) využil fixních poměrů na odříznutí rohů, takže byly všechny rozřezány stejně. Při matematickém zápisu postupuje

Chaikinova metoda následovně: Dostaneme kontrolní polygon $\{P_0, P_1, \dots, P_n\}$, tento kontrolní polygon vylepšíme vygenerováním nové posloupnosti řídicích bodů

$$\{Q_0, R_0, Q_1, R_1, \dots, Q_{n-1}, R_{n-1}\}$$

kde každou novou dvojici bodů Q_i, R_i je třeba brát v poměru $\frac{1}{4}$ a $\frac{3}{4}$ mezi koncovými body segmentu čáry $\overline{P_i P_{i+1}}$.

$$Q_i = \frac{3}{4}P_i + \frac{1}{4}P_{i+1}$$

$$R_i = \frac{1}{4}P_i + \frac{3}{4}P_{i+1}$$

Tyto 2 nové body lze považovat za nový řídicí polygon - vylepšení původního řídicího polygonu. Tento postup lze pro tentýž úsek opakovat, čímž vznikne vyhlazená čára. [10]



Obrázek 1.8: Znázornění Chaikinova algoritmu [9]

2 GTFS

General Transit Feed Specification, zkráceně GTFS, je dokument, který definuje obecný formát pro jízdní řády veřejné dopravy a příbuzné geografické informace. GTFS „feeds“ umožňují veřejným dopravním agenturám zveřejňovat svá přepravní data a vývojářům psát aplikace, které tato data spotřebovávají interoperabilním způsobem neboli vícesystémovým mezinárodním provozuschopným způsobem. [11]

2.1 Historie GTFS

V Portlandu ve státě Oregon v USA se společnost TriMet pokusila jako jedna z prvních řešit problém s plánováním tranzitní dopravy pomocí otevřených dat sdílených širokou veřejností. V roce 2005 společnost TriMet oslovila společnost Google s dotazem, zda mají nějaké plány na začlenění tranzitní dopravy do svých plánovačů výletů na základě veřejných údajů TriMet. Google jim kladně odpověděl a spojily síly s implementací jednoho z prvních plánovačů výletů v Portlandu.

Jedním z prvních problémů, kterým TriMet a Google čelily, byl problém udržitelných dat - pro zajištění kvalitních cest by plánovač cest potřeboval přepravní řád, trasu a údaje o zastávkách v elektronickém formátu, který by byl neustále aktuální. Společnost TriMet ve spolupráci se společností Google naformátovala svá přepravní data do snadno udržovatelného a spotřebního formátu, který lze importovat do Map Google. Tento formát dat přepravy se stal známým jako Specifikace zdroje Google Transit (anglicky Google Transit Feed Specification (GTFS)). V roce 2005 byla tato služba plánování cesty spuštěna jako Google Transit.

Od tohoto roku se GTFS stal nejpopulárnějším datovým formátem pro přepravní služby na světě. Spousta agentur se rozhodla sdílet své GTFS údaje s veřejností, zatímco některé agentury zůstaly zdrženlivé a přístup k datům nechaly jen některým partnerům. Ke 2. prosinci 2019 uvádí OpenMobilityData 1233 poskytovatelů s veřejně přístupnými kanály GTFS, z nichž 465 je ve Spojených státech.

V důsledku inovací vývojářů třetích stran jsou data GTFS nyní využívána různými softwarovými aplikacemi třetích stran k mnoha různým účelům, včetně plánování výletů, map, vytváření jízdních řádů, mobilních dat, vizualizace, přístupnosti, analytických nástrojů pro plánování a informační systémy v reálném čase. V roce 2010

byl název formátu GTFS změněn na General Transit Feed Specification, aby přesně reprezentoval jeho použití v mnoha různých aplikacích mimo produkty Google. [12]

2.2 GTFS dataset

GTFS „feed“ nebo lépe jako GTFS dataset² je ZIP soubor, který obsahuje CSV soubory.

CSV, *Comma-separated values*, v překladu *hodnoty oddělené čárkami*, je jednoduchý souborový formát určený pro výměnu tabulkových dat. Data jsou oddělována „oddělovačem“. Ačkoli podle definice by měl být formát oddělen čárkami, oddělovač může být prakticky cokoli. Nejčastějšími oddělovači jsou právě čárky, středníky nebo mezery. CSV soubor se může upravovat v jakémkoliv textovém editoru.

```

shape_id,shape_pt_lat,shape_pt_lon,shape_pt_sequence,shape_dist_traveled
L991V1,50.07553,14.51539,1,0.00000
L991V1,50.07371,14.51461,2,0.20965
L991V1,50.07361,14.51456,3,0.22173
L991V1,50.07351,14.51451,4,0.23382
L991V1,50.07340,14.51447,5,0.24552
L991V1,50.07329,14.51444,6,0.25817
L991V1,50.07319,14.51441,7,0.26988
L991V1,50.07308,14.51437,8,0.28158
L991V1,50.07298,14.51435,9,0.29298
L991V1,50.07288,14.51433,10,0.30438
L991V1,50.07278,14.51431,11,0.31579
L991V1,50.07269,14.51429,12,0.32623

```

Obrázek 2.1: Ukázka CSV formátu ze souboru shapes.txt

V GTFS datasetu může být v současné době až 17 CSV souborů v textové podobě. Slovem „může“ je myšleno to, že některé CSV soubory jsou požadované či podmíněně požadované a jiné volitelné. Jaké CSV soubory obsahuje dataset záleží na konkrétním dopravním systému, který tento dataset vytváří.

Termín „požadované“ znamená, že daný CSV soubor se musí nacházet v GTFS datasetu nebo dané pole se musí nacházet v CSV souboru a v tomto poli musí být uvedena hodnota pro každý záznam.

Termín „podmíněně požadované“ znamená, že daný CSV soubor nebo pole je vyžadován za určitých podmínek, které jsou uvedeny v popisu souboru nebo pole. Mimo tyto podmínky je tento soubor nebo pole volitelný.

Termín „volitelné“ znamená, že daný CSV soubor nebo pole může být vynecháno. V případě zahrnutí volitelného pole mohou být některé položky prázdné řetězce, což je ekvivalentní s prázdným polem.

²kolekce dat, která by měla být publikována na permanentní URL adrese

V následující tabulce 2.1 jsou přehledně zobrazeny všechny CSV soubory, které v současnosti v GTFS datasetu mohou být.

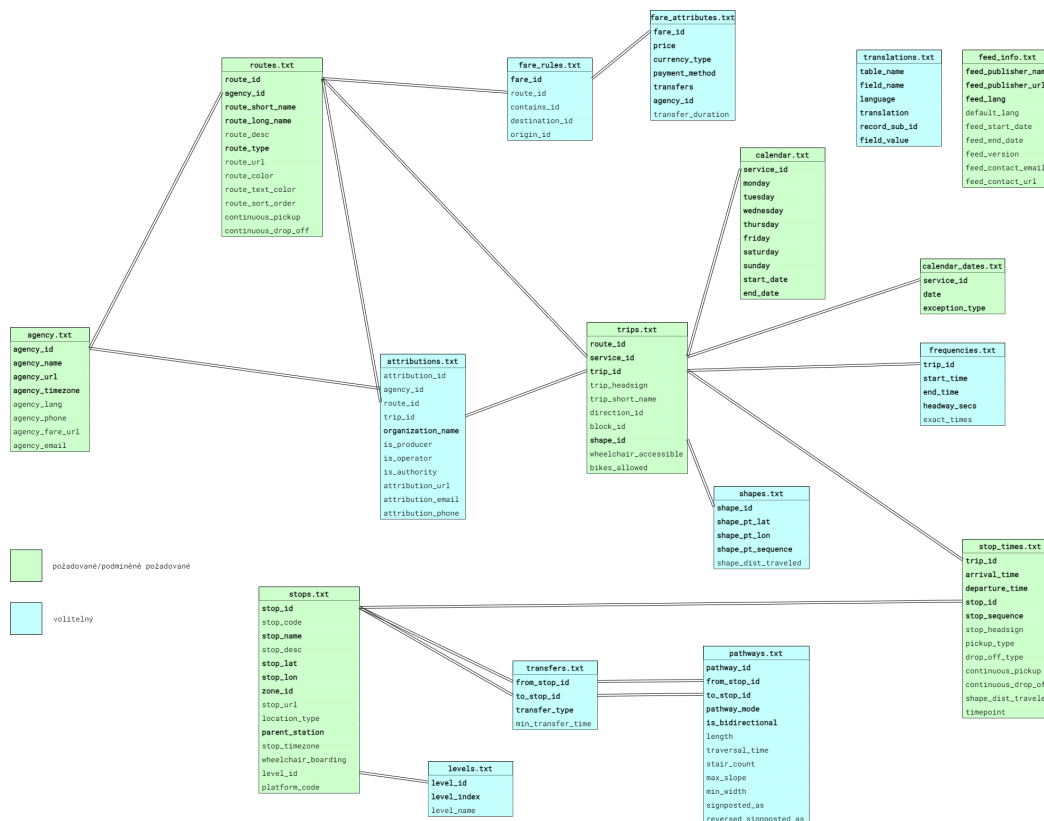
požadované/podmíněně požadované	volitelné
agency.txt	fare_attributes.txt
stops.txt	fare_rules.txt
routes.txt	shapes.txt
trips.txt	frequencies.txt
stop_times.txt	transfers.txt
calendar.txt	pathways.txt
calendar_dates.txt	levels.txt
feed_info.txt	translations.txt
-	attributions.txt

Tab. 2.1: Seznam CSV souborů v GTFS datasetu

Každý CSV soubor v GTFS datasetu má v prvním řádku názvy polí, do kterých je tento soubor rozřazen. Jednotlivá pole mají různé datové typy, které jsou barva, kód měny, datum, email, enum (výčet), ID, kód jazyka, zeměpisná délka, zeměpisná šířka, nezáporné číslo s plovoucí desetinnou čárkou, nezáporné celé číslo, telefonní číslo, čas, text, časové pásmo a URL adresa.

Jedno z nejdůležitějších polí je pole s datovým typem ID, což je hodnota jednoznačně určující každý záznam. Právě tento datový typ umožňuje propojení jednotlivých CSV souborů mezi sebou. ID může být sekvence libovolných UTF-8 znaků. Pole s datovým typem ID se označují na konci názvu s „_id“.

Na následujícím obrázku 2.2 jsou zobrazeny soubory GTFS datasetu pomocí diagramu. Diagram slouží spíše jako schématické zobrazení propojení mezi jednotlivými CSV soubory, které je platné k datu 14.5.2021. Na obrázku je taktéž tučně zobrazeno, které pole v daném CSV souboru jsou požadované, podmíněně požadované nebo volitelné.



Obrázek 2.2: Diagram GTFS datasetu

Pro moji diplomovou práci byly dále důležité datové typy jako zeměpisná délka a zeměpisná šířka, které již podle názvu obsahují zeměpisnou délku a šířku v souřadnicovém systému WGS84, barva zakódovaná jako šestimístné hexadecimální číslo, nezáporné číslo s plovoucí desetinnou čárkou a nezáporné celé číslo nebo enum, což jsou předem definované konstanty.

2.2.1 Soubor stops.txt

Soubor *stops.txt* se skládá ze 14 polí, z čehož 6 polí je požadovaných nebo podmíněně požadovaných a zbytek volitelných. Samotný soubor je požadovaný a měl by se nacházet v každém GTFS datasetu.

Prvním polem je vždy zpravidla *stop_id*, které je požadované a má datový typ ID. Tato hodnota jednoznačně určuje každou zastávku. Pro PID_GTFS, což je GTFS dataset Pražské integrované dopravy, se pole *stop_id* skládá z kombinace písmen a čísel v souvislosti s typem spoje.

Druhým polem je volitelné pole *stop_code* v datovém typu text, což je krátký text nebo číslo, které identifikuje lokaci pro řidiče.

Třetí pole je s datovým typem text *stop_name*. Jak už název napovídá, tak obsahuje název lokace. Je podmíněně požadované kvůli dalšímu volitelnému devátému poli *location_type* s datovým typem enum, které obsahuje druhy lokace. Toto pole je definováno pomocí čtyř konstant:

- 0 nebo prázdné - zastávka nebo nástupiště
- 1 - železniční stanice
- 2 - vchod/východ ze železniční stanice
- 3 - místo ve stanici, které neodpovídá s žádnou z ostatních konstant z *location_type*
- 4 - specifické místo, kde lidé mohou nastoupit/vystoupit z vozidla

Čtvrtým polem je volitelné pole *stop_desc* v textovém datovém typu obsahující popis místa, které poskytuje užitečné a kvalitní informace.

Pátým a šestým polem je *stop_lat* a *stop_lon* s datovým typem zeměpisná šířka a délka obsahující přesně tyto dvě hodnoty. Tyto dvě pole jsou podmíněně požadované kvůli poli *location_type*.

Sedmým polem s datovým typem ID je pole *zone_id*, které je pro tuto diplomovou práci obzvláště důležité. Je podmíněně požadované kvůli CSV souboru *fare_rules.txt*, pokud jsou v něm poskytovány informace o jízdném. Pokud záznam v CSV souboru *stops.txt* představuje stanici nebo vchod do stanice, je *zone_id* ignorováno.

Osmým polem je volitelný *stop_url* s URL adresou na webovou stránku o místě záznamu.

Deváté pole je *location_type* a bylo vysvětleno společně s třetím polem.

Desátým polem je podmíněně požadované pole *parent_station* opět kvůli propojení s polem *location_type*. Má datový typ ID odkazující na pole *stop_id* a definuje hierarchii mezi různými místy definovanými v *stops.txt*. a obsahuje ID nadřazeného umístění.

Poslední čtyři pole jsou volitelné. Prvním z nich je *stop_timezone* s datovým typem časového pásma, *wheelchair_boarding* s datovým typem enum, které označuje, zda je z daného místa možné nastupovat na invalidní vozík. Předposlední je *level_id* s datovým typem ID odkazující na soubor *levels.txt* definující úroveň

umístění zastávky a poslední je *platform_code* s datovým typem text, což je identifikátor platformy pro zastávku patřící stanici.

Všechny tyto pole mají pevné pořadí a nesmí se přeházet. Na obrázku 2.3 je ukázka CSV souboru *stops.txt* pro PID_GTFS dataset, kde nejsou využívány všechny volitelné pole.

```

stop_id,stop_name,stop_lat,stop_lon,zone_id,stop_url,location_type,parent_station,wheelchair_boarding,level_id,platform_code
U50S1,"Budějovická",50.04441,14.44879,"P",,1,,1,,
U52S1,"Chodov",50.03167,14.49096,"P",,1,,1,,
U75S1,"Kolbenova",50.11040,14.51640,"P",,1,,1,,
U78S1,"Ládví",50.12659,14.46945,"P",,1,,1,,
U100S1,"Vltavská",50.10030,14.43849,"P",,1,,1,,
U106S1,"Opatov",50.02789,14.50919,"P",,1,,2,,
U115S1,"Nádraží Holešovice",50.10892,14.44007,"P",,1,,1,,
U118S1,"Flora",50.07828,14.46259,"P",,1,,2,,
U135S1,"Hloubětín",50.10669,14.53639,"P",,1,,1,,
U142S1,"Hlavní nádraží",50.08351,14.43414,"P",,1,,1,,
U157S1,"Bořislavka",50.09834,14.36283,"P",,1,,1,,
U163S1,"Hradčanská",50.09744,14.40316,"P",,1,,2,,
U190S1,"I. P. Pavlova",50.07433,14.43031,"P",,1,,1,,
U209S1,"Jiřího z Poděbrad",50.07772,14.45075,"P",,1,,2,,
U228S1,"Kačerov",50.04184,14.45931,"P",,1,,2,,
U237S1,"Karlovo náměstí",50.07459,14.41698,"P",,1,,2,,
U258S1,"Lužiny",50.04450,14.33044,"P",,1,,1,,
U286S1,"Háje",50.03078,14.52697,"P",,1,,1,,
U306S1,"Nemocnice Motol",50.07539,14.34152,"P",,1,,1,,
U321S1,"Dejvická",50.10047,14.39317,"P",,1,,1,,
U332S1,"Střížkov",50.12617,14.48954,"P",,1,,1,,
U360S1,"Malostranská",50.09172,14.40921,"P",,1,,2,,

```

Obrázek 2.3: Ukázka CSV souboru *stops.txt* pro PID_GTFS dataset

3 Pražská Integrovaná doprava (PID)

Pražská integrovaná doprava (PID) je dopravní systém, který zahrnuje jak pozemní dopravu (tramvaje, železnici, městské a příměstské autobusové linky, lanovou dráhu na Petřín), tak i tu podzemní (metro). Tento dopravní systém zahrnuje i některé přívozy. Systém probíhá integrací společnými přepravními a tarifními podmínkami a jednotným dopravním řešením včetně koordinace jízdnicích řádů. [13]



Obrázek 3.1: Logo Pražské integrované dopravy [13]

3.1 ROPID

Chod integrace systému zajišťuje Regionální organizátor pražské integrované dopravy (zkráceně ROPID), což je příspěvková organizace hlavního města Prahy. Jeho úloha je organizační a kontrolní a ze své práce se odpovídá orgánům samosprávy a státní správy, které jej zabezpečením dopravy pověřily.



Obrázek 3.2: Logo organizace ROPID [13]

ROPID se zabývá vytvářením, rozvíjením a udržováním systému Pražské integrované dopravy v Praze a okolí, včetně návazností na jiné systémy jako jsou Integrovaná doprava Plzeňského kraje, Doprava Ústeckého kraje, Integrovaný dopravní systém Libereckého kraje nebo Integrovaná regionální doprava Královéhradeckého a Pardubického kraje. Taktéž se zabývá vytvářením zásad a standardů dopravní obsluhy a jejich aplikace v závislosti na dostupných finančních zdrojích a jejich projednání s obcemi, okresními úřady a dopravci. ROPID vybírá dopravce, uzavírá

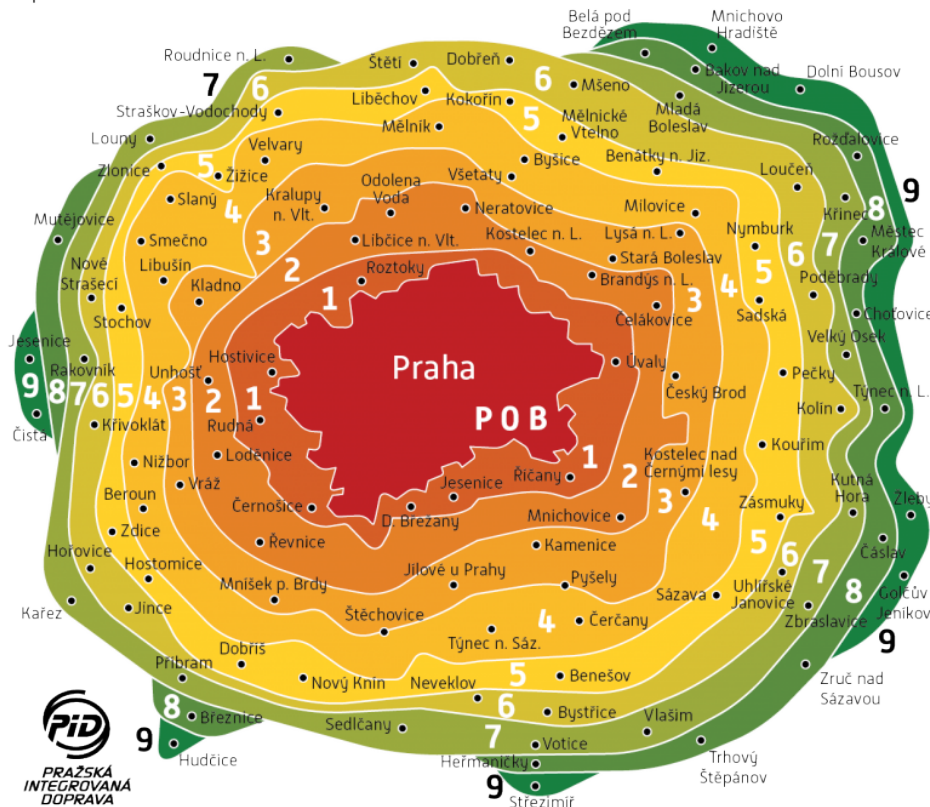
smlouvy o závazku veřejné služby jménem města Prahy k zajištění provozu PID s dotčenými obcemi, Středočeským krajem a dopravci a kontroluje jejich plnění. Náplní organizace ROPID jsou i uspořádání finančních toků v systému PID, návrh tarifů a jízdného v systému PID a zajištění jednotnosti informačního systému PID. [14]

3.2 Tarifní pásma PID

Tzv. „tarifní pásma“ je rozdělení Pražské integrované dopravy do zón, kde v jednotlivých zónách platí rozdílné cenové podmínky. Rozdělení tarifních pásem je následující: P, 0, B, 1, 2, 3, 4, 5, 6, 7, 8, 9. Tarifní pásma P, 0 a B se nacházejí v Praze a územně se spolu překrývají. Ostatní pásma (1 až 9) se z větší části nacházejí ve Středočeském kraji a z menší části v Plzeňském kraji, Ústeckém kraji a na Vysočině.

Zjednodušené schéma tarifních pásem PID

platnost od 1. 10. 2018



Obrázek 3.3: Schéma tarifních pásem PID [13]

Do pásma P jsou zařazeny všechny linky metra, tramvají, městských autobusů, přívozů, včetně lanové dráhy na Petřín a vybraných železničních stanic a zastávek v centru Prahy. Pásmo P má dvojnásobnou tarifní hodnotu (tj. je počítáno jako dvě tarifní pásma). Do pásma 0 jsou zařazeny dojezdové úseky příměstských autobusových linek a vybrané železniční stanice a zastávky v širší oblasti okolo centra Prahy. Do pásma B jsou zařazeny úseky příměstských autobusových linek a vybrané železniční stanice a zastávky v okrajových částech Prahy. [13]

Pro pásma nacházející se ve Středočeském kraji jsou zařazeny jednotlivé stanice a zastávky příměstských autobusových linek PID a vlaků zahrnutých do PID. Příslušnost stanice nebo zastávky k tarifnímu pásmu je vždy dána jízdním řádem konkrétní linky.[13]

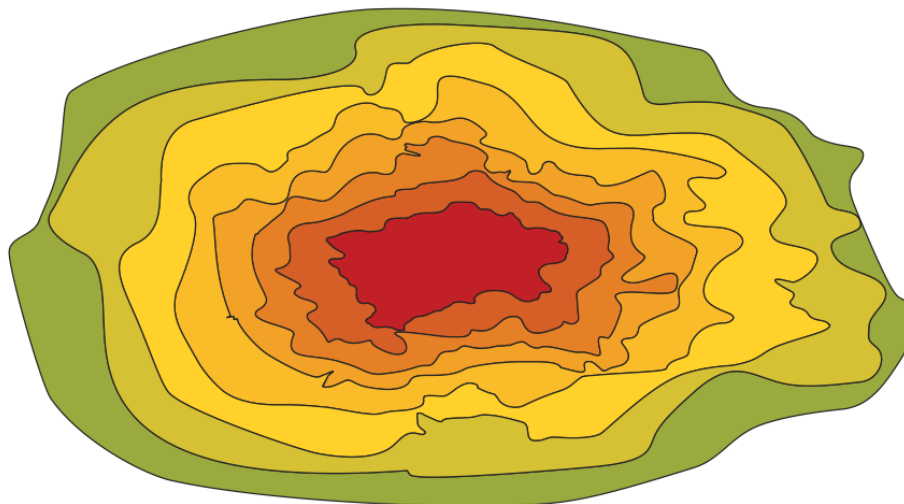
3.2.1 Tvorba tarifních pásem PID

Tarifní pásma Pražské integrované dopravy se tvoří téměř celá manuálně. Zeptal jsem se tedy paní Ing. Zuzany Šaškové z ROPIDu, která se tvorbou tarifních pásem zabývá, jakou metodiku při tom používá.

Paní Ing. Šašková používá na úpravu dat software ArcMap od společnosti Esri. *„Původní myšlenka při tvorbě tarifních pásem byla zohledňovat v mapě zastavěná území obce tak, aby je tarifní pásma pokud možno neprotínala,“* od této myšlenky ustoupila kvůli náročnosti zpracování. *„Dnes mám vytvořeny polygony jednotlivých pásem (nikoliv donuty), každé pásmo ručně upravím dle aktuálních změn.“* V projektu při úpravách pracuje s jednoduchými nástroji, používá symbologii k „obarvení“ zastávek a jednotlivých polygonů tarifního pásma stejnou barvou, hraniční zastávek výraznou barvou případně pro následné úpravy. *„Když ručně vytvaruji všechny polygony tak, aby procházely mezi zastávkami, které mají přiřazené jedno pásmo a skrz zastávky, které mají přiřazená dvě pásma, pustím si skriptu Pythonu, který z polygonů vyřeže donuty jednotlivých tarifních pásem a nakonec je spojí do jedné vrstvy.“*

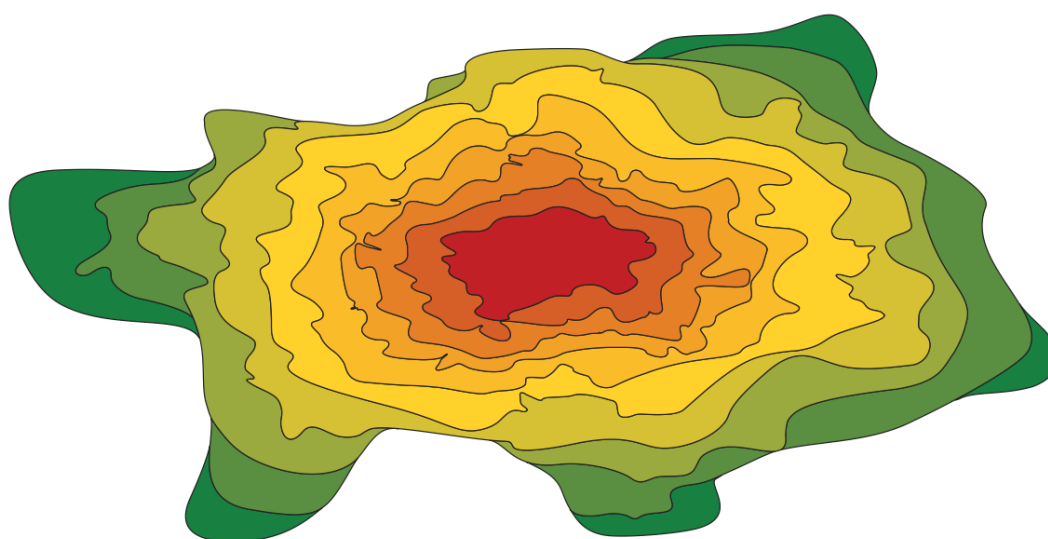
Takto paní Ing. Šašková upravuje tarifní pásma od dubna 2018 při každé zásadnější úpravě zastávek. Zatímco dříve byla důležitá schématická verze mapy, v současnosti se připravuje on-line mapa PID, u které již je zapotřebí, aby tarifní pásma odpovídala realitě a upravovala se v návaznosti na postup integrace dalších oblastí Středočeského kraje do PID.

Zde na obrázku 3.4 je uvedena nejstarší dochovaná podoba pásem z dubna roku 2018. Pro zajímavost si lze všimnout, že v té době tarifní pásmo 7 bylo poslední a všechna pásma tvarově připomínala spíše tu schématickou verzi.



Obrázek 3.4: Nejstarší dostupná verze tarifních pásem PID

Jak již bylo zmíněno v kapitole 1, tak na portálu *Opendata hlavního města Prahy* je pravidelně zveřejňován tvar tarifních pásem. Nejnovější verzi zveřejnila organizace ROPID 8. září 2020. Naopak první verze byla zveřejněna již 16. září 2016, bohužel tato verze nikde nebyla k nalezení. Zde je alespoň zobrazena verze tarifních pásem z 8. září 2020, kde byly pro přehlednost přebrány barvy pásem z schématu od PID (viz 3.3)



Obrázek 3.5: Tarifní pásma zveřejněná organizací ROPID z 8. září 2020

4 Použité technologie

4.1 QGIS

QGIS je profesionální geografický informační systém. Software je zdarma ke stažení a je tzv. open-source (zdrojový kód je veřejně). Zdrojový kód je publikován ve službě GitHub (viz podkapitola 4.4.1). Vývojářem softwaru může být kdokoliv, potvrzovat a kontrolovat změny můžou jen ověření přispěvatelé. Software byl vyvíjen v programovacím jazyku C++ a Python (viz 4.2)



Obrázek 4.1: Logo systému QGIS [15]

QGIS je oficiálním projektem nadace OSGeo (Open Source Geospatial Foundation), což je nevládní nezisková organizace, jejíž cílem je podporovat a prosazovat společný vývoj otevřených geoinformačních technologií. Běží na operačních systémech Linux, Unix, Mac OSX, Windows a Android. Rastrové a vektorové formáty podporované systémem QGIS mohou být uloženy v souborech anebo databázích.

Kromě desktop verze systému QGIS existuje i QGIS Server, který umožňuje publikovat projekty a vrstvy QGIS jako služby WMS, WMTS, WFS a WCS kompatibilní s OGC (Open Geospatial Consortium - mezinárodní standardizační organizace o geoprostorových datech a službách). Dále existuje webový klient, kde lze publikovat QGIS projekty a využívat funkce symboliky a značení. Pro mobilní zařízení tu jsou různé aplikace jako QField, Input a IntraMaps Roam, se kterými také lze používat QGIS projekty a to například přímo v terénu.

QGIS má jako většina softwarů svou dokumentaci, která je sestavena pomocí nástroje Sphinx. Dokumentace je taktéž open-source na službě GitHub a upravovat ji může každý. Obsahuje informace pro běžné desktop uživatele systému QGIS, pro přispěvatele dokumentace nebo pro QGIS vývojáře. V dokumentaci si může uživatel například dočíst informace nebo nápovědu o různých QGIS nástrojích.

Právě QGIS vývojáři potřebují dokumentaci pro vytváření zautomatizovaných skriptů nebo QGIS zásuvných modulů (pluginů). Skripty a větší část pluginů jsou

vytvářena v programovacím jazyku Python (viz 4.2) a menší část v programovacím jazyku C++. Instalaci pluginu lze provést z nabídky uvnitř systému QGIS anebo pomocí ZIP souboru. Pro zobrazení pluginu v QGIS nabídce je potřeba nechat plugin schválit správci systému QGIS, jestli splňuje všechny povinné aspekty QGIS pluginů.

Součástí systému QGIS je i Model Designer (nebo také Graphical modeler, česky Grafický modelář). Pomocí Grafického modeláře lze tvořit modely QGIS nástrojů, což jsou uspořádané nástroje do jednoho řetězového postupu. Pomáhá lépe se v tomto postupu orientovat, měnit vstupy funkcí, pojmenovávat nástroje a jejich parametry, přidávat komentáře či jednotlivé nástroje seskupovat do skupin.

4.2 Python

Python je moderní, dynamický, skriptovací programovací jazyk s veřejným zdrojovým kódem. Je snadný na naučení, lehký čitelný a v současné době velmi populární. Tento programovací jazyk využívají softwaroví inženýři, matematici, datoví analytici, vědci, účetní a síťoví inženýři. Díky jednoduchosti programovacího jazyka Python se využívá k učivu na základních školách.



Obrázek 4.2: Logo Pythonu [16]

Důvodů, proč je tento programovací jazyk tak populární, je mnoho. Python je vyšší programovací jazyk, což znamená lepší srozumitelnost než u nižších programovacích jazyků a programy zapsané ve vyšších jazycích jsou obvykle kratší a lépe čitelné. Další důvodem je multiplatformnost, což znamená, že se můžou aplikace programovacího jazyka Python sestavit a běžet na různých platformách jako Windows, Mac či Linux. Komunita Pythonu je obrovská, což je další výhodou tohoto jazyka. Například jenom v Čechách je mnoho skupin na různých sociálních sítích. Python má také spoustu knihoven, frameworků a nástrojů, které lidem usnadňují programování.

Podporuje různá programovací paradigmatata jako například objektově orientované, imperativní, procedurální nebo funkcionální. Programovacím paradigmatem

se rozumí základní programovací styl, který se liší v pojmech a abstrakcích, které tvoří jednotlivé prvky programu, a krocích, ze kterých se výpočet skládá. [17]

4.3 PyQt

Aby aplikaci vytvořenou v Pythonu mohl ovládat člověk, který se neorientuje v programování a ani nechce, je nutné ke kódu aplikace vytvořit grafické uživatelské rozhraní (GUI). Python má širokou škálu knihoven pro vytváření GUI jako například Tkinter, wxPython, PySide2 nebo PyQt. Právě PyQt je již standardně vestavěný v systému QGIS, tak se zdá jako nejlepší volbou pro tvorbu pluginů.

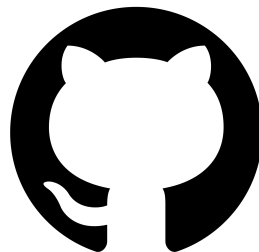
PyQt je vazba Pythonu pro aplikační framework Qt vyvinutý společností RiverBank Computing Ltd. Je k dispozici ve třech verzích: PyQt6 podporující Qt v6, PyQt5 podporující Qt v5 a PyQt4 podporující Qt v4, avšak PyQt4 s Qt v4 již není v nových verzích systému QGIS podporována. PyQt je multiplatformní a lze nainstalovat na Windows, macOS, Linux, iOS a Android. [18]

4.4 Ostatní

Pro verzování kódu anebo pro jeho psaní byly použity i další technologie. Diplomová práce by šla tvořit i bez nich, ale s nimi se zrychloval a usnadňoval její postup.

4.4.1 GitHub

GitHub je služba poskytující internetový hosting pro správu verzí pomocí Git, což je software pro sledování změn v libovolné sadě souborů. Tato služba například poskytuje řízení přístupu, sledování chyb, správu úkolů nebo nepřetržitou integraci. Základní služby poskytuje zdarma, ale za pokročilé funkce je nutné zaplatit.



Obrázek 4.3: Logo GitHubu [19]

4.4.2 PyCharm

PyCharm je vývojové prostředí vyvinuté firmou JetBrains s.r.o. pro programovací jazyk Python. Usnadňuje práci s kódem, zvyšuje produktivitu a zpřehledňuje kód například zvýrazněním syntaxe Pythonu. PyCharm obsahuje širokou kolekci pluginů, které lze doinstalovat a jsou vyvíjeny uživateli. Skrz PyCharm lze propojit i Git, takže psaný kód lze snadno zálohovat. Stejně jako GitHub PyCharm poskytuje základní funkce zdarma, ale profesionální verzi s pokročilejšími funkcemi je nutné si zaplatit.



Obrázek 4.4: Logo PyCharmu [20]

5 Praktická část

5.1 Aktualizace pluginu do verze 1.0

Po dokončení pluginu *GTFS Loader* v předmětu Free Software GIS měl plugin veškerou základní funkcionalitu, kterou měl mít (odkaz na plugin [zde](#)³). Tou bylo načítání GTFS ZIP souboru do systému QGIS, rozbalení ZIP souboru, načtení CSV souborů do geodatabázového formátu GeoPackage (dále GPKG), vytvoření vektorových vrstev pro soubory *stops.txt* a *shapes.txt*, symbologie vektorové vrstvy *shapes* a vložení CSV souborů a vrstev do projektu QGIS.



Obrázek 5.1: Logo pluginu GTFS Loader

Plugin obsahoval nedostatky, které bylo nutné před zveřejněním v QGIS repozitáři vyřešit. Proto byl veškerý proces přesunut na pozadí, aby celý QGIS software „nezamrzal“ a mohly se při jeho výpočtech provádět i jiné akce. To se provedlo díky třídě programovacího jazyka Python *QgsTask* a jejím metodám, které byly zděděny z této třídy. [21]

Pro zobrazování procesu během výpočtu byla použita třída *QProgressBar* a její metody. Zobrazení postupu bylo implementováno do lišty zpráv systému QGIS spolu s chybovými hláškami.



Obrázek 5.2: Ukazatel průběhu (ProgressBar) v liště systému QGIS

5.2 Tvorba vizualizace tarifních pásem (experimentální verze pluginu 2.0)

V následujících podkapitolách je uveden postup tvorby rozšíření pluginu. Nejprve bylo vytvořeno dialogové okno pro potvrzení nebo odmítnutí tvorby tarifních pásem

³<https://github.com/ctu-fgis/2020-b-qgis-gtfs-plugin>

(podkapitola 5.2.1). Dále byl vytvořen hrubý tvar tarifních pásem z rozdílným postupem pro tarifní pásma P, 0 a B a pro tarifní pásma 1 až 9 (podkapitola 5.2.2). Tarifní pásma 1 až 9 byla tvořena se vzájemně oddělenými vektorovými vrstvami. Hrubý tvar nebyl vizuálně vzhledný a k referenčním tarifním pásmům z portálu *Opendata hlavního města Prahy* se nepřibližoval, takže byl zjemněn (podkapitola 5.2.3). Zároveň byla zjemněna i hraniční pásma, což byla nově vytvořená tarifní pásma pro zastávky na hranici mezi pásmy, které se nepodařilo spojit (podkapitola 5.2.4). Hladká tarifní pásma byla následně spojena do jedné vektorové vrstvy (podkapitola 5.2.5) a byla k ní přidána symbologie (podkapitola 5.2.6).

Při průzkumu technologií byla možnost vypracovat tvorbu tarifních pásem pomocí Grafického modeláře, který je zabudovaný v systému QGIS. Kvůli omezeným možnostem úprav nástrojů se z těchto myšlenek upustilo a byl vytvářen postup do prázdného skriptu v programovacím jazyce Python pomocí nástrojů systému QGIS.

Pro použití nástrojů systému QGIS ve skriptu programovacího jazyka Python bylo potřeba importovat modul *processing*, který má funkci *run*, do které se vkládají dva parametry. První parametr je ID nástroje ve formě *řetězce* a druhý je *slovník* vstupních parametrů. Vstupní parametry se lze dozvědět z QGIS dokumentace. [22]

5.2.1 Dialogové okno pro tvorbu tarifních pásem

Aby se uživatel při používání pluginu mohl rozhodnout, či chce vytvářet tarifní pásma nebo ne, bylo vytvořeno dialogové okno pomocí PyQt (viz obrázek 5.3). Toto dialogové okno se vždy zobrazí po stisknutí tlačítka *Load* pro načtení GTFS ZIP souboru. Jak je v závorce napsáno, tvorba tarifních pásem je zatím jen pro GTFS dataset PID_GTFS. Tvorba tarifních pásem pro ostatní GTFS soubory může být jako námět pro další závěrečné práce nebo ho mohou vypracovat uživatelé služby GitHub.



Obrázek 5.3: Dialogové okno s možností tvořit/netvořit tarifní pásma

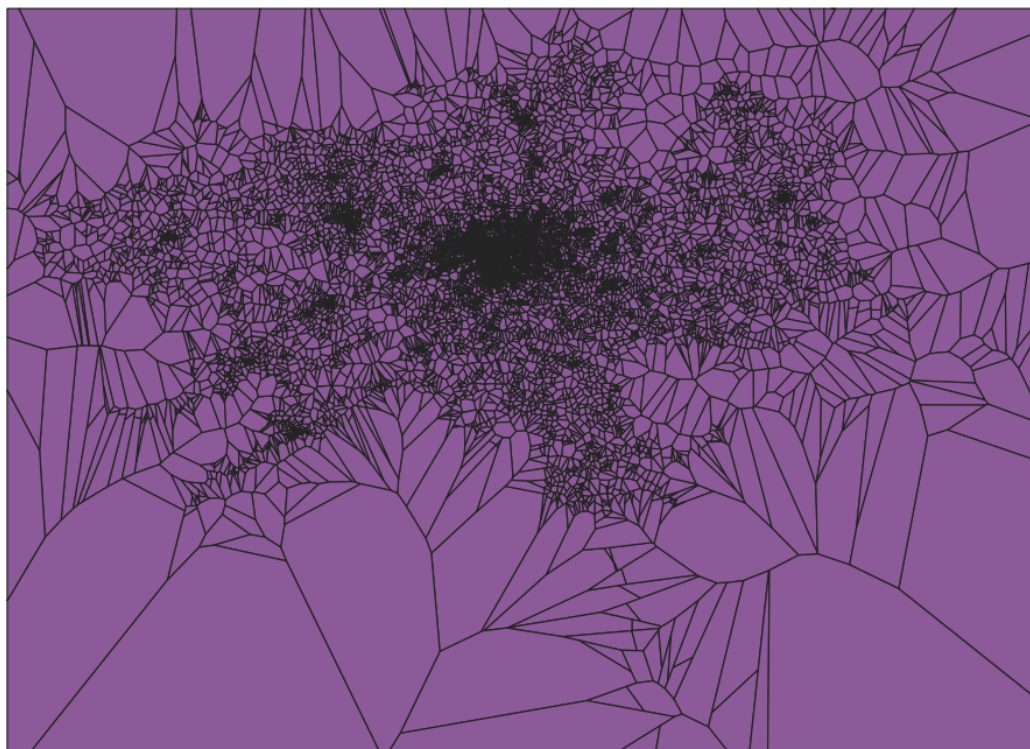
Ovládání dialogového okna je jednoduché. Disponuje běžnými tlačítky pro minimalizaci okna a vypnutí okna, což mimochodem zruší kompletně proces načítání GTFS souboru. Mimo to tu jsou dvě tlačítka *Yes/No*, které buďto povolí vytváření tarifních pásem anebo nepovolí a proces běží jako ve verzi pluginu 1.0.

Kvůli přehlednosti byl založen nový skript *zones.py* v programovacím jazyce Python se třídou *GtfsZones*. Instance této třídy byla vytvořena v hlavním skriptu *GTFS.py*.

5.2.2 Hrubý tvar tarifních pásem

GTFS dataset obsahuje povinný CSV soubor *stops.txt* (viz kapitola 2.2.1). Tento soubor obsahuje mimo jiných polí také pole *zone_id*, které bylo klíčové při tvorbě tarifních pásem. Pole *zone_id* má datový typ *řetězec* a znamená, ve kterém tarifním pásmu daná zastávka leží. Verze 1.0 pluginu *GTFS Loader* soubor *stops.txt* převádí do vektorové vrstvy ve formě bodů. Z této bodové vrstvy byla vytvořena vektorová vrstva Voronoi diagramů nástrojem *Voronoi polygons* v programu QGIS.

Vstupem do tohoto nástroje byla vektorová vrstva *stops* a výstupem byly Voronoi polygony.



Obrázek 5.4: Voronoi polygony pro všechny zastávky

Pro každé tarifní pásmo byly vybrány zastávky pomocí třídy *QgsVectorLayer* a její metody *selectByExpression*, která má parametr *expression* ve formě datového typu *string*. Třída *QgsVectorLayer* zároveň vrací vektorovou vrstvu, která může být považována jako vstup do nástroje QGIS.

Tarifní pásma P, 0 a B

Speciálně pro tarifní pásma na území Prahy (tarifní pásma P, 0 a B) byl tvořen rozdílný postup než pro ostatní pásma ležící ve Středočeském kraji a okolí. Do metody *selectByExpression* byl vložen parametr s dotazem:

$$"zone_id" \text{ IN } ('P','0','B') \text{ AND } "location_type" = 0$$

Po proběhnutí tohoto dotazu byly vybrány všechny zastávky ležící v tarifních pásmech P, 0 a B.

V zadávaném výrazu figuruje taktéž údaj o poli *location_type*, což je typ lokace. Hodnota nuly (nebo prázdná hodnota) je právě lokace zastávky viz 2.2.1.

Některé zastávky se nacházely na stejném místě a měly duplicitní geometrii, což vytvářelo v dalším zpracování problém s duplicitními polygony. Aby se takové věci ve výsledku zamezilo, byl použit nástroj *Delete duplicate geometries*, který ve vstupní vektorové vrstvě našel duplicitní geometrie a smazal je.



Obrázek 5.5: Vybrané Voronoi polygony pro tarifní pásma P, 0 a B

Pro vybrané zastávky bylo potřeba vybrat ty Voronoi polygony, které svou pozicí dané zastávky protínaly. To bylo provedeno nástrojem *Select by location*, do kterého vstupovala vrstva vybraných zastávek a vrstva Voronoi polygonů. Výsledkem tohoto nástroje byla vektorová vrstva vybraných Voronoi polygonů.

Tyto polygony byly následně nástrojem *Dissolve* spojeny do jednoho společného polygonu. Vstupem tohoto nástroje byl výstup nástroje *Select by location* a výstupem byl polygon s jednou geometrií.



Obrázek 5.6: Výsledek nástroje Dissolve pro tarifní pásma P, 0 a B

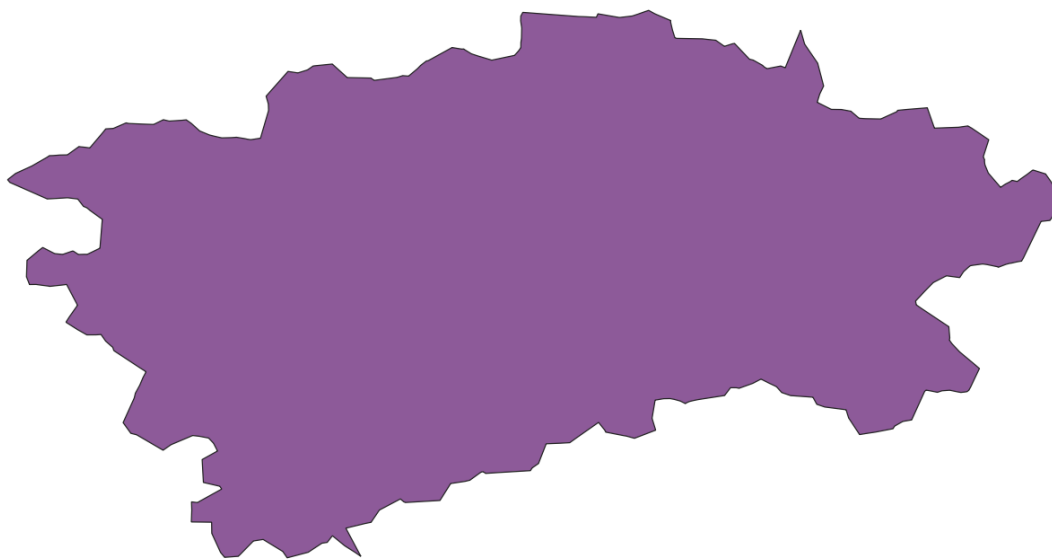
Kvůli mnoha zastávkám, které v oblasti pásem P, 0 a B v poli *zone_id* obsahovaly hodnotu *NULL*, vznikla uvnitř polygonu spousta děr. V těchto dírách někdy ležely části malých polygonů, které po nástroji *Dissolve* tvořily jeden celistvý polygon. A proto nebylo možné rovnou použít nástroj *Delete holes*, nýbrž nejprve nástroj *Multipart to singleparts*.

Nástroj *Multipart to singleparts* rozdělil jeden vícedílný polygon na jednotlivé menší polygony. Vstupem do tohoto nástroje byla vektorová vrstva výstupu nástroje *Dissolve* a výstupem byla vektorová vrstva s rozděleným polygonem.

Poté se pomocí metody *selectByExpression* třídy *QgsVectorLayer* vyhledal největší polygon. Výraz vkládaný jako parametr do metody *selectByExpression* byl následující:

$$\$area = maximum(\$area, "zone_id")$$

Takový polygon byl uložen do nové vektorové vrstvy.



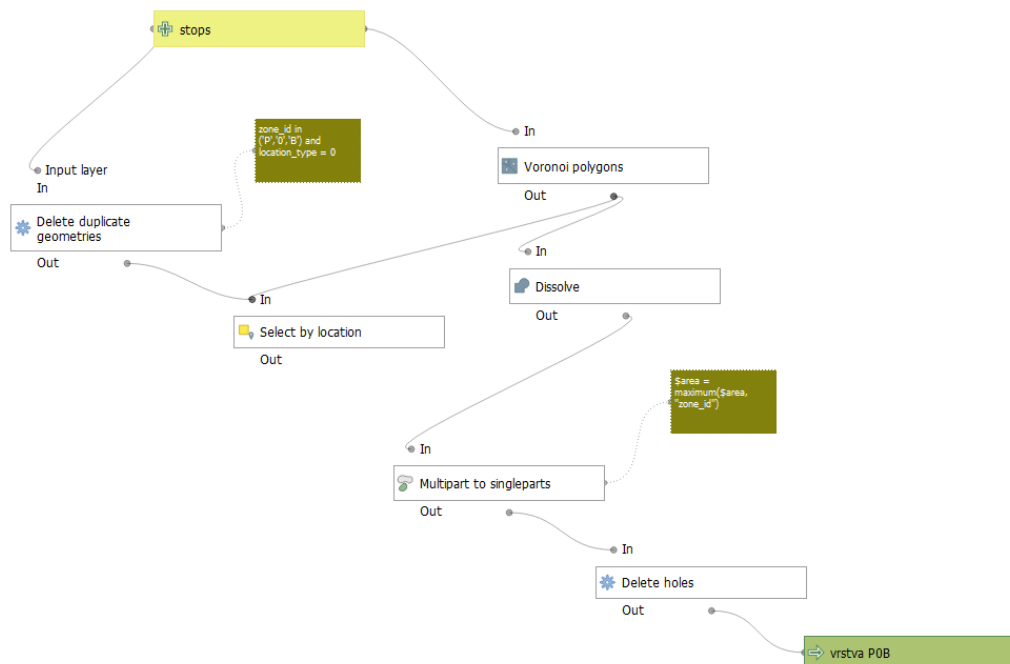
Obrázek 5.7: Výsledek nástroje Delete holes pro tarifní pásma P, 0 a B

Poté byl použit nástroj *Delete holes*, který zaplnil vytvořené díry. Vstupem do tohoto nástroje byla vektorová vrstva s největším polygonem z nástroje *Dissolve*. Nástroj obsahoval volitelný parametr rozlohy, který byl nastaven na hodnotu 500, se kterým nástroj *Delete holes* byl řízen a byly zaplněny rozlohově menší díry.

Znázornění tohoto postupu je zobrazeno v obrázku 5.8. Toto zobrazení bylo provedeno v Grafickém modeláři.

Tarifní pásma 1 až 9

Obdobný postup jako u tarifních pásem P, 0 a B, avšak znatelně jednodušší, byl proveden za pomoci for cyklu pro tarifní pásma 1 až 9. V něm už nebylo nutné použít nástroj pro mazání děr, rozdělávání polygonů na vícedílné geometrie nebo mazání duplicitních bodů.



Obrázek 5.8: Postup pro tarifní pásma P, 0 a B

Zastávky, které ležely v určitém pásmu, byly vyhledány pomocí metody *selectByExpression* třídy *QgsVectorLayer*. Pro zastávky tarifních pásem 1 až 6 to bylo provedeno pomocí dotazu:

`"zone_id" <= "IDpasma" AND "zone_id" != '-' AND "location_type" = 0`

Pro zastávky tarifních pásem 7 až 9 pomocí dotazu:

`"zone_id" = "IDpasma" AND "location_type" = 0`

V dotazu pro tarifní pásma 1 až 6 byly vybrány ty zastávky, které mají hodnotu pole *zone_id* menší jak aktuální hodnota, hodnota pole *zone_id* nebyla rovna „pomlčce“ (jelikož zastávky s touto hodnotou jsou mimo tarifní pásma) a hodnota pole *location_type* byla rovna nule. V dotazu pro tarifní pásma 7 až 9 byly vybrány ty zastávky, které mají hodnotu pole *zone_id* aktuální hodnotě a hodnota pole *location_type* byla rovna nule.

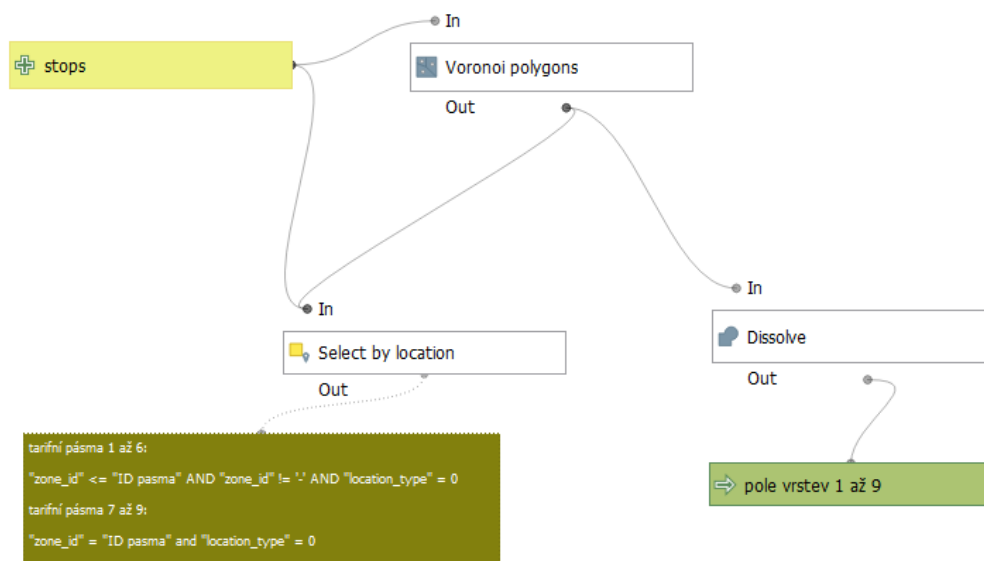
Hlavním důvodem bylo „zalepení“ děr, které vznikaly u vyhlazení tvaru tarifních pásem v kapitole 5.2.3.

Důsledkem dělení těchto dotazů byla časová náročnost délky průběhu následujícího nástroje *Select by location*, kde kvůli operátoru " \leq " u dotazu pro zastávky tarifních pásem 1 až 6 do nástroje vstupovaly všechny zastávky, které byly hodnotou ID menší jak aktuální hodnota. Tím byl pro každé tarifní pásmo násobně zvětšen počet zastávek, které vstupovaly do nástroje *Select by location*.

Pro tarifní pásma 7 až 9 byla časová náročnost stejná jako u tarifních pásem P, 0 a B.

Následně byl pouze spuštěn nástroj *Dissolve* pro spojení Voronoi polygonů, což bylo vstupem do tohoto nástroje, do jednoho celistvého polygonu, což bylo výstupem tohoto nástroje.

V následujícím obrázku 5.9 je souhrnně zobrazen postup pro tarifní pásma 1 až 9.



Obrázek 5.9: Postup pro tarifní pásma 1 až 9

5.2.3 Vyhlazení tvaru tarifních pásem

Pro vytvoření kartograficky přesných a vizuálně přijatelnějších tarifních pásem dosavadní postup nestačil. V souboru *stops.txt* datasetu PID_GTFS existovaly takové zastávky, které měly tarifní pásmo sdílené na rozhraní pásem. Těmito zastávkami měla podle pravidel organizace ROPID tarifní pásma procházet.

Bylo nutné navrhnout další průběh úpravy dosavadního tvaru tarifních pásem, který takovou podmínku zahrnuje a zároveň vyhlazuje tvar polygonů pro lepší vizuální tvar.

Ze spojených polygonů byla vygenerována vektorová vrstva bodů nástrojem *Extract vertices*, která představovala vrcholy spojených polygonů. Pro tuto vrstvu bylo vstupem výstup nástroje *Dissolve* a výstupem byla vektorová vrstva bodů doplněná o pole (mimo původních polí z vektorové vrstvy *stops*) jako *vertex_index*, *vertex_part*, *vertex_part_ring*, *distance* a *angle*. Hodnoty těchto polí nebyly využity v dalším výpočtu.



Obrázek 5.10: Výsledek nástroje *Extract vertices* pro tarifní pásma P, 0 a B

Dále byly s pomocí třídy *QgsVectorLayer* a její metody *selectByExpression* vybrány z původní vektorové vrstvy *stops* ty zastávky, které ležely na hranici tarifních pásem. Takové zastávky obsahovaly v poli *zone_id* hodnoty tarifních pásem oddělené čárkami. Například pro zastávky mezi pásmem 1 a 2 byla hodnota pole *zone_id* uvedena jako „1,2“.

Pro tarifní pásma P, 0 a B byl použit dotaz:

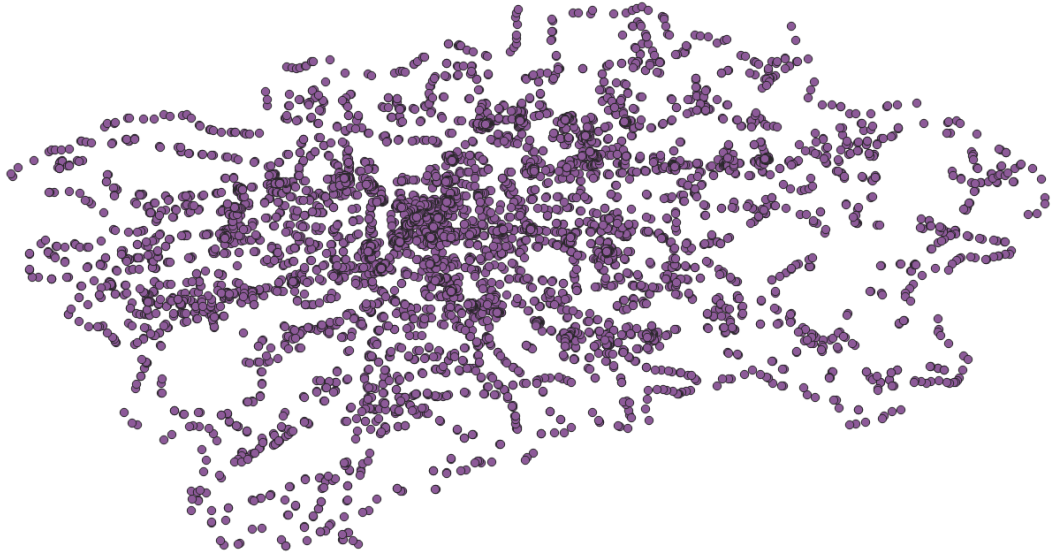
```
"zone_id" NOT IN ('P', '0', 'B', '1', '2', '3', '4', '5', '6', '7', '8', '9')
AND ("zone_id" LIKE '%P%' OR "zone_id" LIKE '%0%'
OR "zone_id" LIKE '%B%')
```

Pro tarifní pásma 1 až 9 byl použit dotaz:

"zone_id" LIKE "ID pásma i, ID pásma i + 1"

Vybraná tarifní pásma byla uložena do nové vektorové vrstvy pro následné a pozdější použití viz kapitola 5.2.4.

Následně pro každé tarifní pásmo nástrojem *Merge vector layers* byly spojeny tři vektorové vrstvy: hraniční zastávky, zastávky uvnitř tarifního pásma a body z výstupu nástroje *Extract vertices*. Tyto zmiňované vektorové vrstvy byly vstupem do tohoto nástroje a výstupem byla vektorová vrstva spojených bodů.



Obrázek 5.11: Spojené body třech vektorových vrstev pomocí nástroje *Merge vector layers* pro tarifní pásma P, 0 a B

Pro spojení bodů a vytvoření jakési obálky byl nejdříve vyzkoušen nástroj *Minimum bounding box* s možností *Convex Hull*, což je aplikace algoritmu Konvexní obálky viz kapitola 1. Tento způsob však velmi zjednodušoval tvar tarifních pásem, a proto nástroj *Minimum bounding box* nebyl použit.

Byla zvolena další možnost nástroje systému QGIS spojující body do určité obálky. Tím nástrojem byl *Concave hull (alpha shapes)* využívající aproximaci *alpha-shapes* viz kapitola 1.

Vstupem nástroje *Concave hull (alpha shapes)* byla vrstva spojených bodů, což byl první parametr nástroje. Dalším parametrem byl práh (*Threshold*) s datovým typem *číslo*, který byl volen od 0 do 1, kde 0 znamenala maximum konkávní obálky

a 1 konvexní obálka. Po několika testovacích spuštěních byla určena hodnota 0,09 jako nejlepší hodnotou pro tvorbu tarifních pásem. Dalším parametrem bylo Povolení děr (Allow holes) s datovým typem *boolean*, který byl nastaven na *False*. Posledním parametrem bylo Rozdělit vícedílnou geometrii na jednotlivé části (Split multipart geometry into singlepart geometries) taktéž s datovým typem *boolean*, který byl nastaven na *True*. Výstupem byla polygonová vektorová vrstva konkávní obálky.

Vytváření konkávní obálky bylo dalším důvodem pro dělení atributových dotazů v kapitole 5.2.2 pro zaplnění děr v tarifních pásmech 6 a nižších.

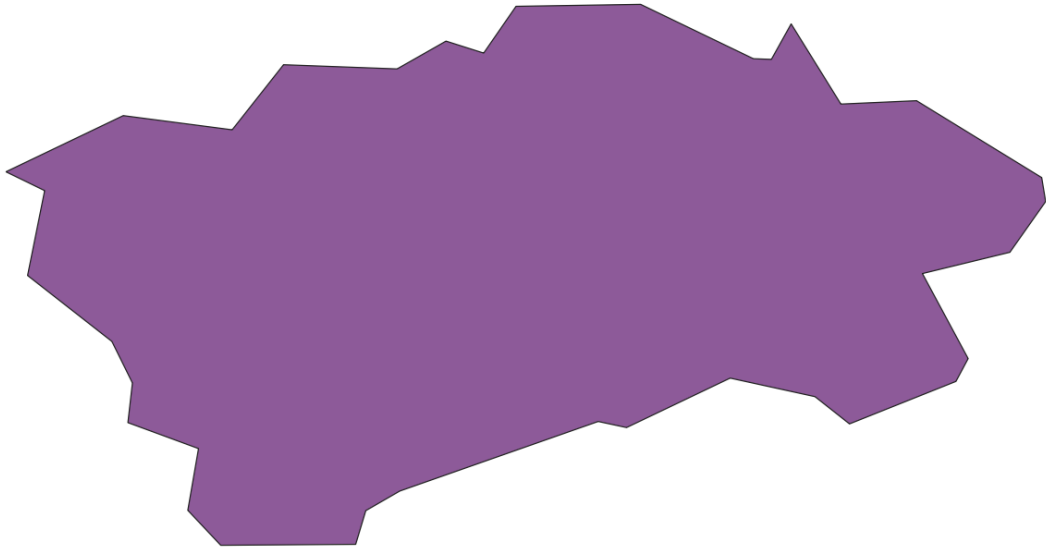


Obrázek 5.12: Konkávní obálka pro tarifní pásma P, 0 a B

Z konkávní obálky byla pomocí nástroje *Simplify* „zjednodušena“ geometrie polygonové vrstvy. Tento nástroj využíval tři druhy zjednodušení vektorové vrstvy: založené na vzdálenosti (algoritmus „Douglas-Peucker“), založené na ploše (algoritmus „Visvalingam“) a přichytávání geometrií k mřížce.

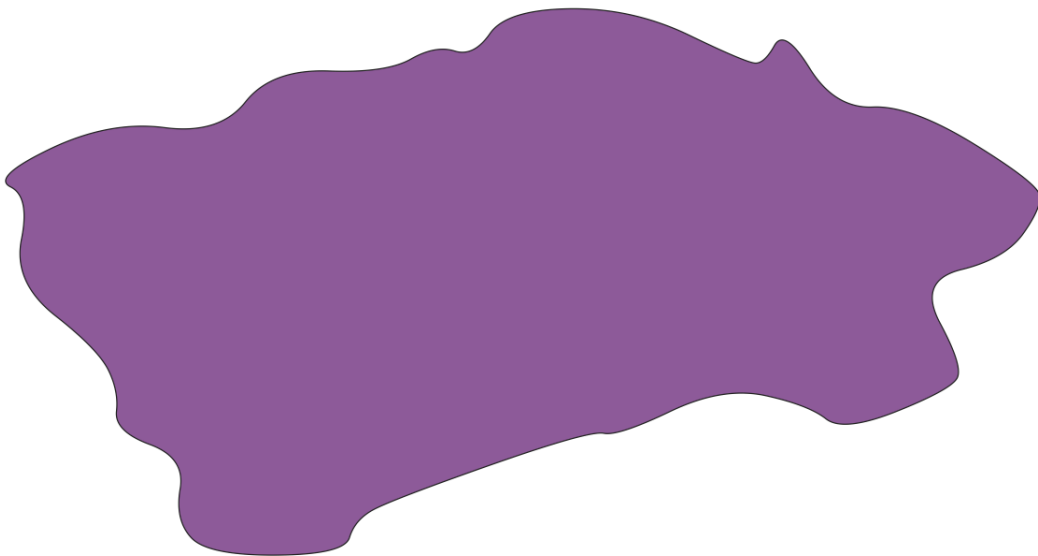
Pro můj postup byla zvolena první volba zjednodušení, pomocí algoritmu „Douglas-Peucker“ viz kapitola 1.

Do nástroje *Simplify* byl vstupem výsledek nástroje *Concave hull (alpha shapes)*, parametrem byl výběr algoritmu a nastavení hodnoty tolerance vzdálenosti. Výstupem byla polygonová vektorová vrstva.



Obrázek 5.13: Výsledek nástroje Simplify pro tarifní pásma P, 0 a B

Výstup nástroje *Simplify* byl vstupem do nástroje *Smooth*. Tento nástroj vyhlazuje geometrie liniové nebo polygonové vrstvy pomocí Chaikinova algoritmu viz kapitola 1.

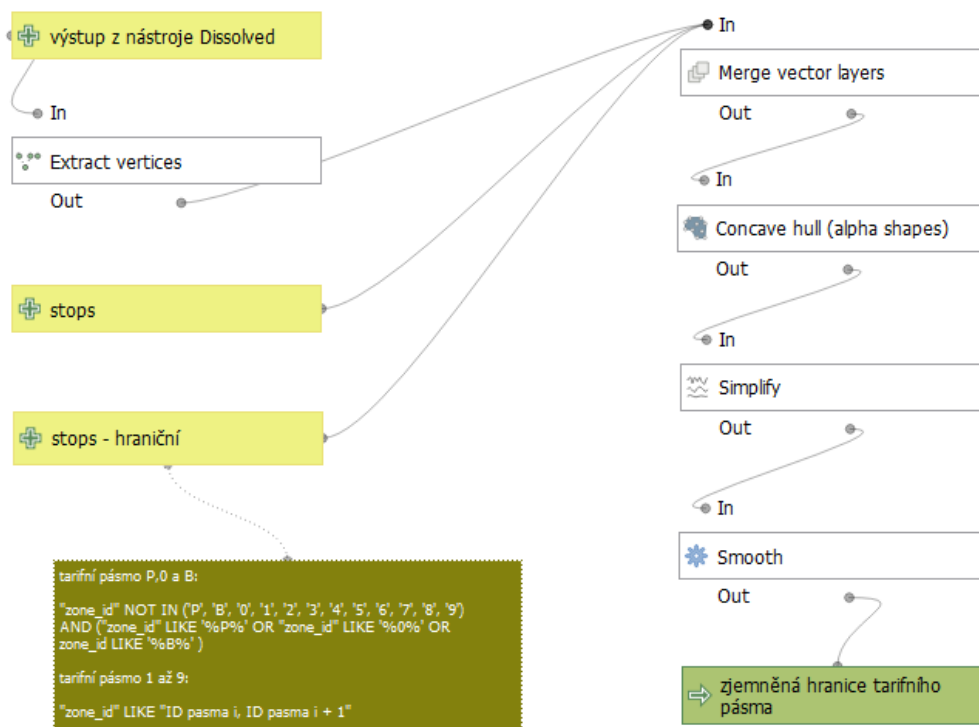


Obrázek 5.14: Výsledek nástroje Smooth pro tarifní pásma P, 0 a B

U nástroje *Smooth* byly zvoleny tři parametry - počet iterací, offset a maximální úhel. Počet iterací znamená, kolik vyhlazovacích iterací bude použito pro každou geometrii. Hodnota počtu iterací byla nastavena na 10, což byla maximální volitelná hodnota (pro co největší hladkost). Parametr offset znamená, jak „těsně“ vyhlazené geometrie sledují původní geometrie. Zde byla ponechána výchozí hodnota 0,25.

Poslední parametr maximálního úhlu lze použít k zabránění vyhlazení uzlů s velkými úhly. Zde byla také ponechána výchozí hodnota 180°. Výstupem nástroje byla polygonová vektorová vrstva.

Na obrázku 5.15 je souhrnně zobrazen postup této části pomocí Grafického mod- eláře.



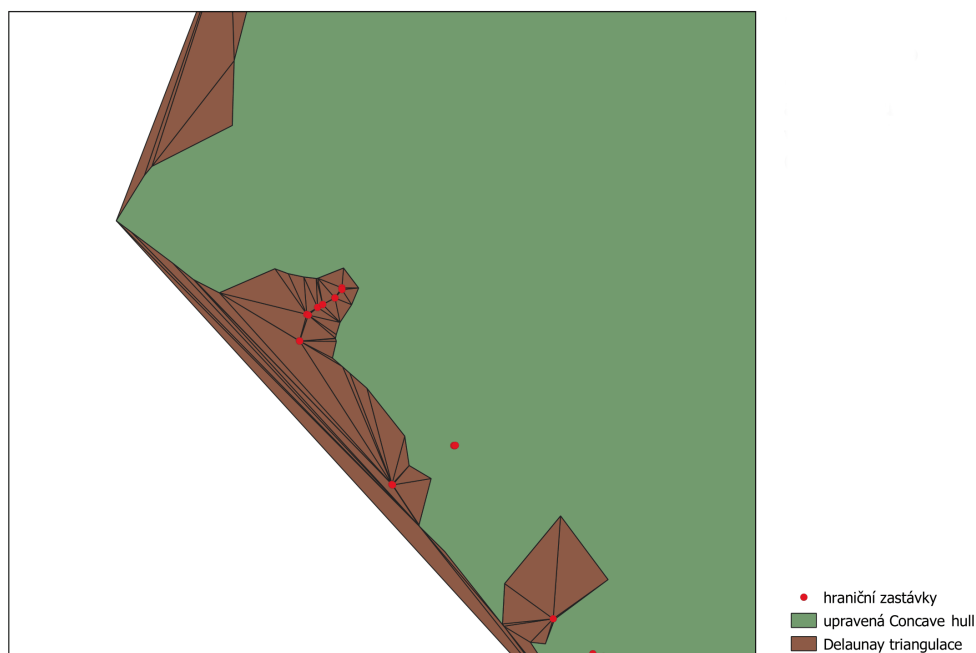
Obrázek 5.15: Postup pro zjemnění tvaru tarifních pásem

5.2.4 Hraniční tarifní pásma

Jedním z největších problémů diplomové práce bylo vypořádat se s hraničními zastávkami. V dosavadním postupu nebyl jejich problém efektivně vyřešen kvůli konkávní obálce, která nezvládala spojovat určité body, které byly požadovány.

Naskytl se tedy myšlenka upravit algoritmus nástroje *Concave hull (alpha shapes)*, který je veřejně publikován ve službě GitHub jako ostatně celý zdrojový kód systému QGIS, k obrazu svému, aby hraniční zastávky propojoval.

Zdrojový kód nástroje byl napsán Piotrem Pociaskem v květnu 2014. Skládá se mimo jiné z nástroje *Delaunay triangulation*, který využívá pro tvorbu Delaunay triangulaci viz kapitola 1, nástroje *Dissolve*, nástroje *Delete holes* a algoritmu *alpha shapes*.



Obrázek 5.16: Ukázka upraveného nástroje Concave hull (alpha shapes) pro část tarifních pásem P, 0 a B

Tento algoritmus maže ty trojúhelníky z Delaunay triangulace, které mají nejdelší stranu menší jak α (volitelný parametr nástroje *Concave hull (alpha shapes)*) násobek maximálního obvodu trojúhelníku. K těmto trojúhelníkům byly přidány i ty trojúhelníky, které mají vrchol společný s jakýmkoliv hraničním bodem.

Výsledek byl oproti očekávání. Místo propojení hraničních zastávek s konkávní obálkou vznikly mezery, které nemohly být dál použity a tento postup byl zavržen.

Od pracovníků z organizace ROPID mi byl navrhnout postup, jak tento problém alespoň částečně vyřešit. Pro hraniční zastávky se stejnou hodnotou $zone_id$, které leží ve své vzájemné blízkosti, vytvořit vlastní tarifní pásmo. Takový postup byl mnohem lépe uchopitelnější, než ten předchozí uvedený, který nebyl povedený.

Uložená vektorová vrstva hraničních zastávek z kapitoly 5.2.3 byla s vektorovou vrstvou Voronoi polygonů použita s pomocí nástroje *Select by location*. Z vektorové vrstvy Voronoi polygonů byly vybrány právě ty polygony, které se svým umístěním protínaly s hraničními zastávkami. To bylo provedeno zvlášť pro každé zastávky na hranici tarifních pásem.

Poté byly nástrojem *Dissolve* seskupeny Voronoi polygony do jednoho polygonu a jedné geometrie. Vstupem tohoto nástroje byly vybrané Voronoi polygony pomocí nástroje *Select by location* a výstupem byla vektorová vrstva seskupených polygonů.

Toto seskupení bylo následně zrušeno pomocí nástroje *Multipart to singleparts*, kde vstupem byla vektorová vrstva seskupených polygonů s jednou geometrií a výstupem byla vektorová vrstva s více geometriemi podle jejich počtu ve vektorové vrstvě.

Důvodem tohoto kroku byl úmysl spočítat počet zastávek, které ležely na území každého polygonu, poté určit hranici součtu zastávek a vymazat ty polygony, které ji nespĺňovaly.

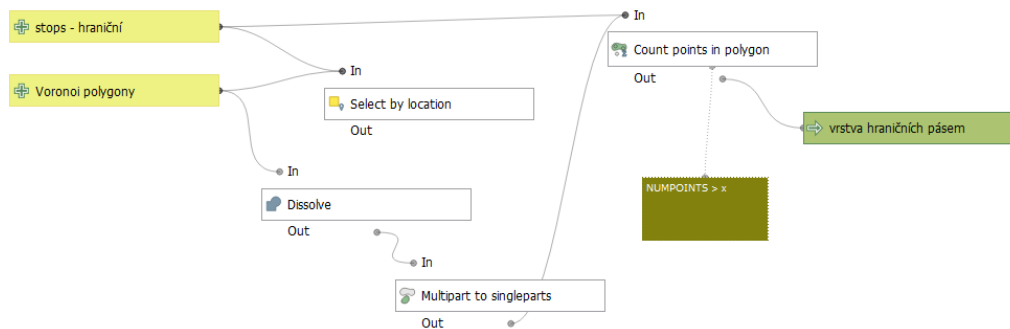
Pro spočítání zastávek byl použit nástroj *Count points in polygon*. Vstupem byla vektorová vrstva rozdělených polygonů nástrojem *Multipart to singleparts* a vektorová vrstva hraničních zastávek. Mimo jiných volitelných parametrů se v nástroji nacházel i parametr *Count field name*, který určoval název nového pole s datovým typem celého čísla, který přibyl na výstupu nástroje. Výchozí název tohoto parametru *NUMPOINTS* byl ponechán. Kromě nového pole vypadala výstupní vektorová vrstva stejně jako vstupní vektorová vrstva polygonů.

Poté podle atributového dotazu bylo rozhodnuto, jaké polygony z vektorové vrstvy polygonů vybrat do nové vektorové vrstvy. Tento atributový dotaz vypadal následovně:

$$NUMPOINTS > X$$

Hodnota čísla X byla nastavena na 5. Byly vybrány ty polygony, ve kterých leželo více jak 5 zastávek.

Tento postup byl proveden for cyklem pro všechna tarifní pásma společně s vyhlazením hranic polygonů, jak je uvedeno v kapitole 5.2.3. Na obrázku 5.17 je v Grafickém modeláři zobrazen přehledný postup pro hraniční tarifní pásma.



Obrázek 5.17: Postup pro hraniční tarifní pásma

5.2.5 Složení tarifních pásem

V dosavadním postupu byla ukládána většina polygonových vektorových vrstev do GPKG jako oddělené. To bylo potřeba změnit z důvodu lepší manipulace s výslednou vrstvou a také kvůli zmenšení velikosti GPKG kvůli ušetření paměti.

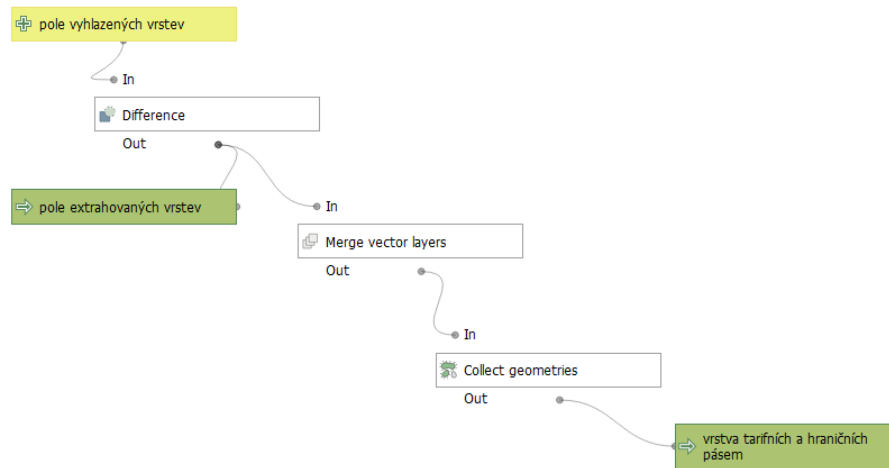
Vektorové vrstvy byly od sebe odečteny nástrojem *Difference*, aby nebyly prolínány. Prvním parametrem tohoto nástroje byla vektorová vrstva, která překrývala druhou vektorovou vrstvou. Druhým parametrem byla vektorová vrstva, která byla překrývána. Výstupem byla odečtená vrstva. To bylo provedeno for cyklem pro všechna tarifní pásma.

Odečtené vektorové vrstvy tarifních pásem byly poté spojeny do jedné vektorové vrstvy pomocí nástroje *Merge vector layers*. Díky tomuto nástroji byly spojeny i hraniční pásma zatím odděleně od tarifních pásem.

Výsledek spojení obsahoval více menších polygonů se stejnými hodnotami *zone_id*. Takové polygony bylo potřeba spojit do jednoho vícedílného polygonu pro každou hodnotu *zone_id*. To bylo provedeno pomocí nástroje *Collect geometries*. Vstupem do tohoto nástroje byla vektorová vrstva z nástroje *Merge vector layers* a pole, podle které se to mělo spojit, bylo nastaveno na *zone_id*. Výstupem byla vektorová vrstva vícedílných polygonů. Tento postup byl proveden i pro hraniční pásma.

Poté zbývalo spojit vektorovou vrstvou tarifních pásem a vektorovou vrstvou hraničních pásem. Kvůli vyhnutí se překrývání vektorových vrstev byl nejdříve použit nástroj *Difference* a následně nástroj *Merge vector layers*.

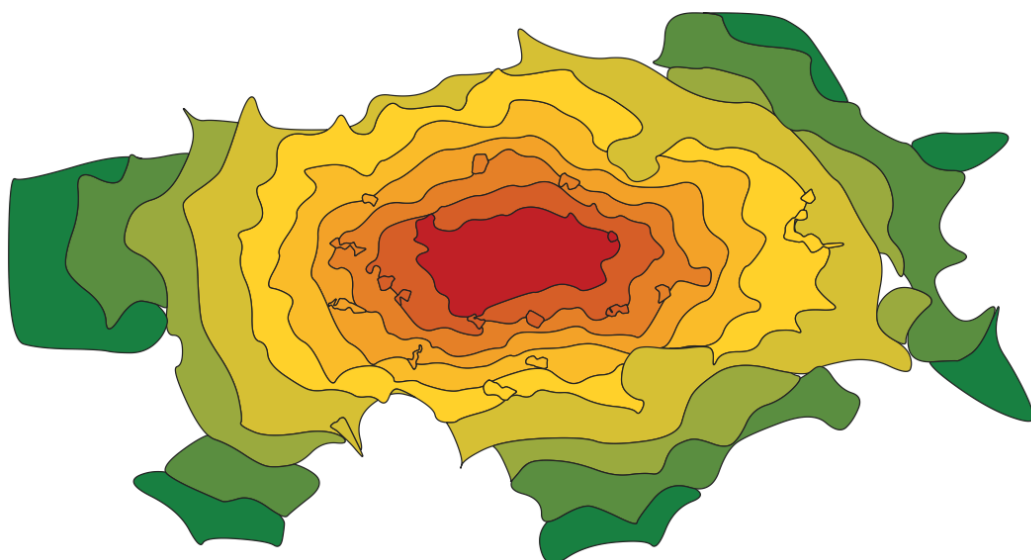
Výsledkem byla jedna vektorová vrstva obsahující polygony tarifních i hraničních pásem. Na následujícím obrázku je přehledně zobrazen postup složení v Grafickém modeláři.



Obrázek 5.18: Postup pro složení vektorových vrstev tarifních pásem

5.2.6 Symbologie tarifních pásem

Pro lepší vizuální vzhled výsledných tarifních pásem byla na konec procesu plug-inu přidána funkce pro obarvování pásem. To bylo provedeno díky instanci třídy *QgsSymbol* a jejím metodám *defaultSymbol* a *setColor* a taktéž díky instancím tříd *QgsRendererCategory* a *QgsCategorizedSymbolRenderer*.



Obrázek 5.19: Výsledný tvar tarifních pásem

Barvy, kterými jsou výsledná tarifní pásma obarvena a které jsou zobrazeny na obrázku 5.19, byly převzaty ze schématu tarifních pásem Pražské integrované dopravy (viz 3.3).

Obrázek výsledného tvaru tarifních pásem byl vytvořen za pomoci PID_GTFS datasetu publikovaný 8. září 2020 na portálu *Opendata hlavního města Prahy*. Ve stejný den byly zveřejněny na stejném portálu i referenční tarifní pásma. Když byl porovnán výsledek tarifních pásem z PID_GTFS datasetu a referenční tarifní pásma, tak se navzájem příliš nepodobaly. Zřejmě to bylo způsobeno tím, že při tvorbě referenčních tarifních pásem byly použity jiné zastávky, než které byly publikovány s datasetem na portálu *Opendata hlavního města Prahy*.

Závěr

Primárním cílem této diplomové práce bylo rozšířit plugin systému QGIS o automatické vytváření exaktního tvaru tarifních pásem Pražské integrované dopravy, jehož referenční podoba je publikována veřejně organizací ROPID. Podkladovými daty byl GTFS dataset taktéž zveřejňovaný organizací ROPID. Mezikrokem k primárnímu cíli bylo zveřejnění pluginu do repozitáře QGIS. Sekundárním cílem bylo vytvořit schématický model pro informativní účely, který by se tolik neshodoval s referenční podobou tarifních pásem.

Mezikrok nutný pro pokračování vývoje byl splněn ve formě aktualizace na verzi pluginu 1.0. V této aktualizaci byly opraveny některé chyby, doplněno uživatelské rozhraní a proces pluginu přeložen na pozadí. Následně byl publikován v oficiálním repozitáři systému QGIS s názvem *GTFS Loader*. K 16. 5. 2021 bylo v záložce *Plugins* uvedeno 420 unikátních stáhnutí pluginu.

Implementace tvorby tarifních pásem byla vytvářena jako experimentální verze pluginu 2.0. Výsledek byl tvořen jako volitelný výstup pluginu ve formě vektorové vrstvy se symbologií jednotlivých tarifních pásem. Výsledný průběh tarifních pásem byl podobný oficiální verzi publikované veřejně organizací ROPID, ale byl doplněn o tzv. hraniční tarifní pásma, která pokrývala větší seskupení hraničních zastávek. Pro ostatní nepokryté hraniční zastávky se nepodařilo najít způsob, jak je napřímo propojit s hranicemi tarifních pásem.

Jak již bylo zmíněno, proces pluginu pro vytváření tarifních pásem nezvládá správně zpracovat hraniční zastávky. Zároveň v oblastech, kde jsou umístěny zastávky velmi vzácně, vznikají „díry“ mezi tarifními pásmy. Tyto „díry“ se ve snaze pro lepší tvar eliminovaly, ale kvůli tomu vznikl další problém překrývajících se tarifních pásem. Každopádně pro lepší tvar je potřeba manuální editace vektorové vrstvy.

Zaměstnanci organizace ROPID bylo dáno pravidlo o nekřížení se intravilánu obce s hranicemi tarifních pásem. Toto pravidlo mělo mít za následek vyhýbání se hranic tarifních pásem s většími zastavěnými oblastmi. Později bylo kvůli obtížnosti s uskutečněním tohoto pravidla upuštěno na přání ze strany pracovníků z organizace ROPID. Současně pro realizaci tohoto pravidla by bylo potřeba načítat

vektorové vrstvy dotčených obcí České republiky, což by razantně zvýšilo datovou náročnost pluginu.

Postup navržený v rámci diplomové práce není rozhodně ideální a je na něm co vylepšovat. Předmět úpravy a vylepšení pluginu se může stát dalším tématem semestrálních či závěrečných prací. Zdrojový kód je již nyní veřejně přístupný v repositáři pluginu s názvem *qgis-gtfs-plugin* ve službě GitHub (odkaz: https://github.com/ctu-geoforall-lab/qgis-gtfs-plugin/tree/pid_zones) a kdokoliv k němu může přidávat návrhy na vylepšení.

Seznam zkratek

CSV	Hodnota oddělená čárkami (Comma-separated values)
DPZ	Dálkový průzkum Země
GIS	Geografický informační systém (Geographic information system)
GPKG	Formát dat pro geografický informační systém (GeoPackage)
GUI	Grafické uživatelské rozhraní (Graphical user interface)
ID	Hodnota jednoznačně určující každý záznam
URL	Jednotná adresa zdroje (Uniform Resource Locator)
WCS	Web coverage service
WFS	Web feature service
WMS	Webová mapová služba (Web map service)
WMTS	Služba webových mapových dlaždic (Web Map Tile Service)
ZIP	Formát pro kompresi a archivaci dat

Odkazy

- [1] Esri. Shapefiles. [online], [cit. 2021-04-12]. Dostupné z: <https://doc.arcgis.com/en/arcgis-online/reference/shapefiles.htm>.
- [2] Tomáš Bayer. Rovinné triangulace a jejich využití. [online], [cit. 2021-04-12]. Dostupné z: <https://web.natur.cuni.cz/~bayertom/images/courses/Adk/adk5.pdf>.
- [3] Tomáš Vaníček. Triangulace. [online], [cit. 2021-04-27]. Dostupné z: https://kix.fsv.cvut.cz/~vanicek/vyuka_z09/Triangulace.ppt.
- [4] Brian D. Corner. Illustration of greedy triangulation. [online], [cit. 2021-04-27]. Dostupné z: https://www.researchgate.net/figure/Illustration-of-greedy-triangulation_fig21_228574112.
- [5] Příspěvatelé Wikipedie. Delaunay triangulation. [online], [cit. 2021-04-27]. Dostupné z: https://upload.wikimedia.org/wikipedia/commons/d/db/Delaunay_circumcircles_vectorial.svg.
- [6] F. Bernardini and C. Bajaj. *Sampling and reconstructing manifolds using alpha-shapes*. Dept. Comput. Sci., Purdue Univ., West Lafayette, 1997. [cit. 2021-04-28].
- [7] Tran Kai Frank Da. Cgal 5.2.1 - 2d alpha shapes. [online], [cit. 2021-04-28]. Dostupné z: https://doc.cgal.org/latest/Alpha_shapes_2/index.html.
- [8] Tomáš Bayer. Voronoi diagram. vlastnosti, použití, konstrukce. zobecněné voronoi diagramy. [online], [cit. 2021-03-23]. Dostupné z: <https://web.natur.cuni.cz/~bayertom/images/courses/Adk/adk6.pdf>.
- [9] Tomáš Bayer. Kartografické generalizační algoritmy generace bodových prvků/skupin. generalizace liniových prvků/skupin. generalizace plošných prvků/skupin. [online], [cit. 2021-04-08]. Dostupné z: <https://web.natur.cuni.cz/~bayertom/images/courses/Adk/adk8.pdf>.
- [10] Kenneth I. Joy. Chaikin's algorithms for curves. [online], [cit. 2021-04-11]. Dostupné z: <https://www.cs.unc.edu/~dm/UNC/COMP258/LECTURES/Chaikins-Algorithm.pdf>.

- [11] Google Developers. Gtfs static overview. [online], [cit. 2021-03-11]. Dostupné z: <https://developers.google.com/transit/gtfs/>.
- [12] TransitWiki users. General transit feed specification. [online], [cit. 2021-04-12]. Dostupné z: https://www.transitwiki.org/TransitWiki/index.php/General_Transit_Feed_Specification.
- [13] Pražská integrovaná doprava. O systému|pražská integrovaná doprava. [online], [cit. 2021-03-11]. Dostupné z: <https://pid.cz/o-systemu/>.
- [14] Příspěvatelé Wikipedie. Ropid. [online], [cit. 2021-04-18]. Dostupné z: <https://cs.wikipedia.org/wiki/ROPID>.
- [15] QGIS Development Team. Qgis. [online], [cit. 2021-04-16]. Dostupné z: <https://www.qgis.org/>.
- [16] Python Software Foundation. Python. [online], [cit. 2021-04-16]. Dostupné z: <https://www.python.org/>.
- [17] Příspěvatelé Wikipedie. Programovací paradigma. [online], [cit. 2021-04-16]. Dostupné z: https://cs.wikipedia.org/wiki/Programovac%C3%AD_paradigma.
- [18] Riverbank Computing. What is pyqt? [online], [cit. 2021-04-22]. Dostupné z: <https://riverbankcomputing.com/software/pyqt>.
- [19] Inc. GitHub. Github. [online], [cit. 2021-05-06]. Dostupné z: <https://github.com/>.
- [20] JetBrains s.r.o. Pycharm. [online], [cit. 2021-05-06]. Dostupné z: <https://www.jetbrains.com/pycharm/>.
- [21] QGIS. Třída qgstask. [online], [cit. 2021-03-25]. Dostupné z: <https://qgis.org/pyqgis/3.16/core/QgsTask.html#qgis.core.QgsTask>.
- [22] QGIS. Qgis dokumentace. [online], [cit. 2021-03-29]. Dostupné z: <https://docs.qgis.org/3.16/en/docs/index.html>.

- [23] Opendata Praha. Pražská integrovaná doprava - tarifní pásma. [online], [cit. 2021-04-12]. Dostupné z: <https://opendata.praha.eu/dataset/ipr-prazska-integrovana-doprava--tarifni-pasma>.
- [24] Boris Odehnal. Some steps of chaikin's algorithm and limit curve. [online], [cit. 2021-04-27]. Dostupné z: https://www.researchgate.net/figure/Some-steps-of-Chaikins-algorithm-and-limit-curve_fig3_258027659.

A Dodatek

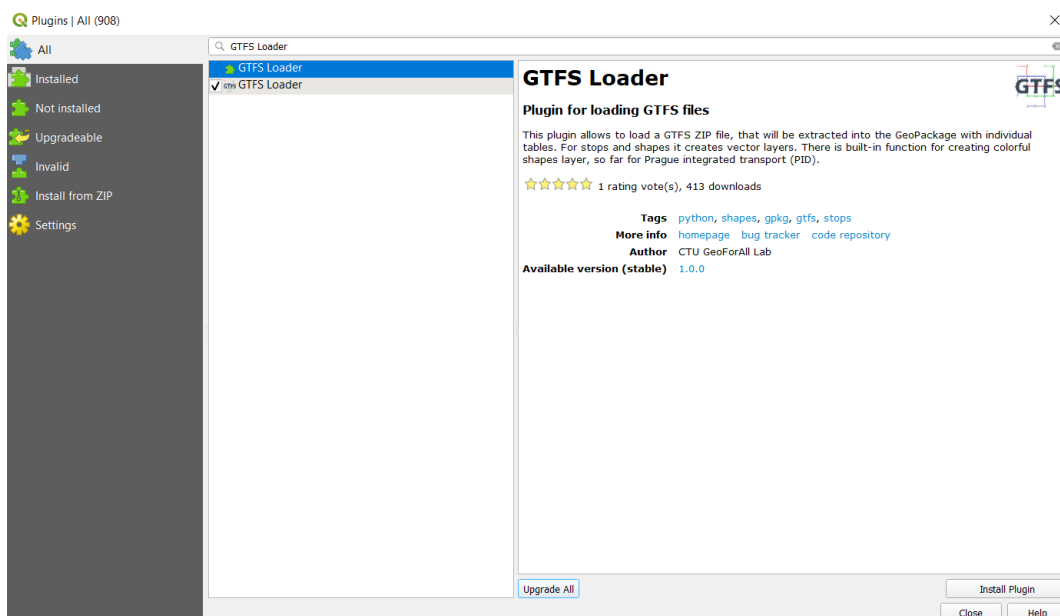
A.1 Instalace pluginu

Popis instalace pluginu je uveden pro software QGIS nastavený v anglickém jazyce.

A.1.1 Instalace pluginu pomocí QGIS repozitáře

Pro nainstalování pluginu *GTFS Loader* v softwaru QGIS se nejprve vybere přes menu panel *Plugins* a poté z nabídky panelu se vybere možnost *Manage and Install Plugins...* Dále se otevře dialogové okno pro výběr instalace pluginu z QGIS repozitáře, zadá se název pluginu a stiskne se tlačítko *Install Plugin*.

V QGIS repozitáři je aktuálně uložena verze pluginu 1.0, která nedisponuje vytvářením tarifních pásem.

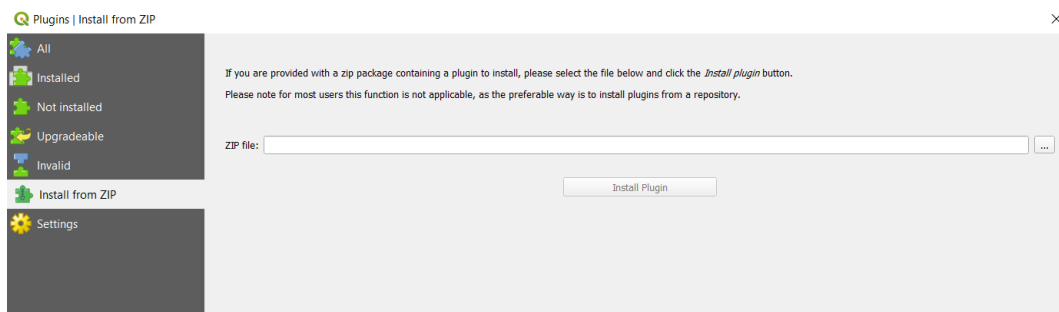


Obrázek A.1: Dialogové okno pro výběr instalace pluginu z QGIS repozitáře

A.1.2 Instalace pluginu pomocí ZIP souboru

K nainstalování experimentální verze pluginu 2.0 je třeba zvolit jiný postup. Z uvedeného odkazu https://github.com/ctu-geoforall-lab/qgis-gtfs-plugin/tree/pid_zones, který směřuje na repozitář pluginu *GTFS Loader* ve službě GitHub, je potřeba stáhnout komprimovaný ZIP soubor. To se provede tak, že se rozklikne tlačítko *Code* a následně se vybere možnost *Download ZIP*.

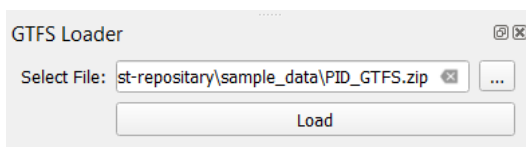
Pro načtení ZIP souboru v softwaru QGIS je potřeba otevřít stejné dialogové okno jako při instalaci pomocí QGIS repozitáře, ale následně se vybere možnost *Install from ZIP*. Do uvedeného pole se zadá celá cesta ke staženému ZIP souboru a poté se stiskne tlačítko *Install Plugin*.



Obrázek A.2: Dialogové okno pro výběr ZIP souboru

A.2 Použití pluginu pro vytvoření tarifních pásem

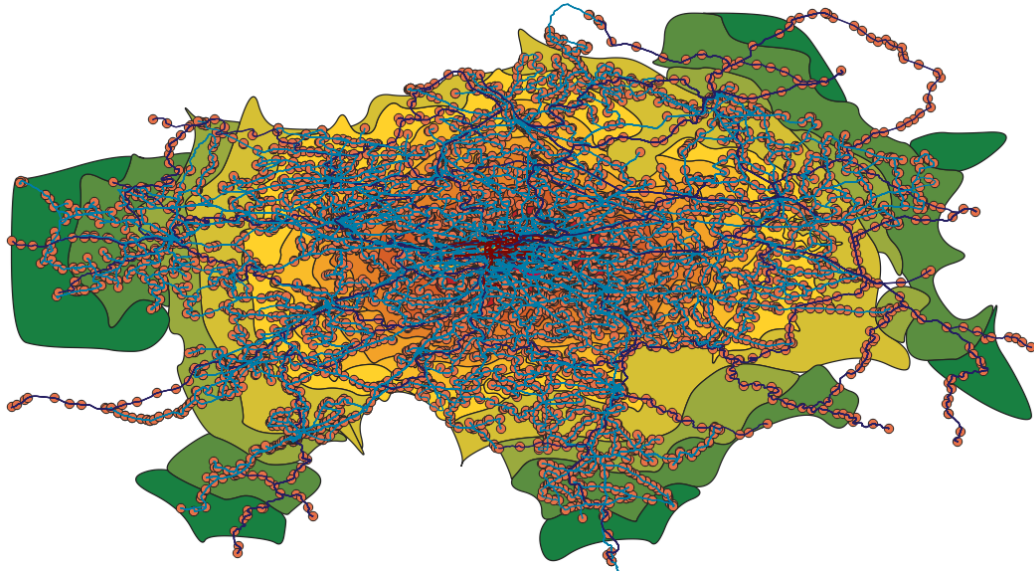
Po instalaci se objeví plugin *GTFS Loader* v panelu *Plugins* anebo jako ikona pluginu v panelu nástrojů. Po stisknutí ikony se v levém dolním rohu objeví dialogové okno pluginu.



Obrázek A.3: Dialogové okno pluginu GTFS Loader

Do pole s popiskem *Select File* se musí vyplnit kompletní cesta k GTFS ZIP souboru (pro vytváření pásem pouze k PID_GTFS ZIP souboru). Dále stačí stisknout tlačítko *Load*. Následně se zobrazí další dialogové okno, zda uživatel chce vytvořit tarifní pásma. Stiskne se *Yes* a proces pluginu začne běžet na pozadí. Délka času tvorby závisí na výkonu počítače. Během procesu uživatel vidí informace o progresu v ukazateli průběhu (ProgressBar).

Po načtení dat se v mapovém okně zobrazí kromě vrstvy *shapes* a *stops* i vrstva *zones*. Na obrázku A.4 je vrstva *zones* jako nejnižše položená. Tyto vrstvy jsou se stejným názvem uloženy ve skupinách v seznamu vrstev.



Obrázek A.4: Výsledek procesu pluginu