VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
FACULTY OF MECHANICAL ENGINEERING

ÚSTAV AUTOMATIZACE A INFORMATIKY
INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

MODERNÍ METODY ŘÍZENÍ SVĚTELNÉ KŘIŽOVATKY

MODERN METHODS FOR INTERSECTION SIGNALING CONTROL

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. Pavel Bartoš
AUTHOR

VEDOUCÍ PRÁCE — Ing. Jakub Kůdela, Ph.D.
SUPERVISOR

BRNO 2022

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadání diplomové práce

Ústav:	Ústav automatizace a informatiky
Student:	**Bc. Pavel Bartoš**
Studijní program:	Aplikovaná informatika a
řízeníStudijní obor:	bez specializace
Vedoucí práce:	**Ing. Jakub Kůdela, Ph.D.**
Akademický rok:	2021/22

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijníma zkušebním řádem VUT v Brně určuje následující téma diplomové práce:

Moderní metody řízení světelné křižovatky

Stručná charakteristika problematiky úkolu:

Práce se bude zabývat moderními metodami pro řízením světelné křižovatky. Tyto metody se dělí jak podle typu algoritmu pro řízení, tak podle toho, na jaký typ křižovatky jsou určeny a jaké jsou zvolena kritéria, který by řízení mělo splňovat. Hlavním cílem práce bude implementace vybranémetody pro zvolený typ křižovatky.

Cíle diplomové práce:

Popsat problém řízení světelné křížovatky.
Provést rešerši moderních metod pro tento problém. Pro vybranou metodu vytvořit software implementaci.

Seznam doporučené literatury:

LI, X, SUN, J.-Q. Turning-lane and signal optimization at intersections with multiple objectives. Engineering Optimization. 2019, 51 (3), 484-502.

LIST, G. F, CETIN, M. Modeling Traffic Signal Control Using Petri Nets. IEEE Transactions onIntelligent Transportation Systems. 2004, 5 (3), 177-187.

CHEN, L., ENGLUND, C. Cooperative Intersection Management: A Survey. IEEE Transactions onIntelligent Transportation Systems. 2016, 17 (2), 570-586.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2021/22

V Brně, dne

L. S.

doc. Ing. Radomil Matoušek, Ph.D. ředitel ústavu	doc. Ing. Jaroslav Katolický, Ph.D. děkan fakulty

ABSTRAKT

Tato diplomová práce se zabývá optimálním řízením světelné křižovatky. V úvodu jsou zavedeny pojmy a představeny metody řízení. Následně je zvolen software SUMO od společnosti Eclipse, který slouží k vytvoření modelu. Po vzoru reálné křižovatky jsou modelovány jak rozložení a logika, tak dopravní toky. Samotné dopravní toky jsou částečně náhodné. Řízení je prováděno za pomoci tří délek zelených front, vyhodnocování probíhá porovnáváním průměrné a maximální čekací doby. Vybírají se tak Pareto optimální body. Prvním algoritmem pro získání optimálního nastavení je prohledávání v mřížce. Druhým algoritmem byl implementován NSGA-II. Nastavení se porovnávají jak mezi sebou, tak s ohledem na nastavení a časové okno.

ABSTRACT

This diploma thesis deals with the optimal control of traffic lights at intersections. In the introduction, concepts are introduced and control methods are presented. Subsequently, Eclipse's SUMO software is chosen to create the model. Following the model of a real intersection, both layout and logic and traffic flows are created. The traffic flows itself are partially randomized. The control is performed using three green queue lengths and the evaluation is done by comparing the average and maximum waiting times. Pareto optimal points are thus selected. The first algorithm to obtain the optimal setting is a grid search. The second algorithm has been implemented by NSGA-II. The settings are compared with each other as well as with respect to the settings and the time window.

KLÍČOVÁ SLOVA

SUMO, Simulace městské dopravy, řízení světelné křižovatky, prohledávání v mřížce, optimální řízení dopravy, genetický algoritmus, NSGA-II

KEYWORDS

SUMO, Simulation of urban mobility, traffic light control, Grid search, optimal traffic control, genetic algorithm, NSGA-II

ÚSTAV AUTOMATIZACE A INFORMATIKY

2022

BIBLIOGRAFICKÁ CITACE

BARTOŠ, Pavel. *Moderní metody řízení světelné křižovatky*. Brno, 2022. Dostupné také z: https://www.vutbr.cz/studenti/zav-prace/detail/140076. Diplomová práce. Vysoké učení technické v Brně, Fakulta strojního inženýrství, Ústav automatizace a informatiky. Vedoucí práce Jakub Kůdela.

PODĚKOVÁNÍ

Chci poděkovat rodině za zázemí potřebné k práci. Dále vedoucímu za dobré rady, časovou flexibilitu i nadhled. V neposlední řadě své přítelkyni za silnou oporu.

ČESTNÉ PROHLÁŠENÍ

Prohlašuji, že, že tato práce je mým původním dílem, vypracoval jsem ji samostatně pod vedením vedoucího práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury.

Jako autor uvedené práce dále prohlašuji, že v souvislosti s vytvořením této práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následku porušení ustanovení § 11 a následujících autorského zákona c. 121/2000 Sb., včetně možných trestně právních důsledků.

V Brně dne 20. 5. 2022 .....

Bc. Pavel Bartoš

OBSAH

1 ÚVOD

Zvláště v dnešní době je kladen čím dál větší důraz na propustnost naší dopravní sítě. S tím, jak bohatne společnost, dochází ke zvyšování koncentrace vozidel v populaci. Není nutně řeč pouze o osobních autech, v popularitě rostou též vozidla jednostopá, ať už motorová či nikoli. S výjimkou velkých měst, kde je dopravní infrastruktura vysoce rozvinutá i v ohledu osobní dopravy, je vlastnění vozidla stále pro většinu lidí investice převyšující její náklady.

Aby dokázala silniční síť pojmout tuto zvyšující se poptávku, dochází k jejím úpravám na několika úrovních současně. Jednak probíhá snaha o vytvoření obchvatů, čímž se zvyšuje průtok dopravy v okolí velkých uzlů a ulehčuje jim. Jednak je snaha vytvářet víceproudé komunikace, na kterých je možné obsloužit větší množství vozidel. Obě tato opatření však mají jedno společné, a to nutnost vyhrazení prostoru. Stavba obchvatu vyžaduje administrativně i finančně náročné operace odkupu pozemků, stejně tak jako rozšiřování komunikací nad rámec jejích původně zamýšlených plošných výměr.

Pokud se jedná o ryze městskou dopravu, tedy stav, kdy vyžadujeme průjezd libovolným množstvím městských uzlů, jsou obě tyto metody téměř nepoužitelné. V husté zástavbě je ve většině případů další rozšiřování komunikací nemožné a objížďky jsou vhodné pouze tehdy, když nejsme nuceni přímo vjet do města. V takových prostředích se pak primárně optimalizuje dopravní průtok v rámci nastavení logiky systému. Řeč je o dopravních pravidlech, a to na všech úrovních.

Různé systémy libovolné složitosti, obsahující síť tepen či magistrál, jsou vidět v každém větším městě. Problém optimálního nastavení pravidel těchto systémů však zásadně trpí vlastností podtrhovanou v urbanizovaných oblastech. Dopravní tok libovolným městským uzlem je silně dependentní na časovém okně. Tento trend lze pozorovat na všech prvcích dopravní infrastruktury. Silniční síť tak dokáže paralyzovat či alespoň silně ovlivnit časovou náročnost jejího průjezdu. Navíc v mnohých případech dochází k asymetrii dopravní zátěže. Stavu, kdy špičky poptávkových toků jednotlivých spojnic křižovatky jsou odděleny v čase. Typickým příkladem je ranní zhuštění dopravy ve směru do města a odpolední zvýšená koncentrace směrem z města.

Bez nutnosti složité výstavby další infrastruktury se nabízí ovládání dopravních pravidel jako hlavní nástroj zvýšení kýženého dopravního průtoku primárně v inkriminovaných dobách dopravních špiček. Nástrojem adaptivního řízení jsou v tomto případě světelné křižovatky, jelikož značky, nejsou-li umístěny na digitálních informačních tabulích, nelze efektivně měnit v čase. Hlavním nastavitelným parametrem v těchto křižovatkách je délka zelené fronty, tedy doba svitu zelené barvy pro každou spojnici zvlášť.

Simulačním prostředím pro testování jednotlivých nastavení bylo zvoleno Eclipse SUMO, Simulation of Urban Mobility (Simulace městské dopravy). Jedná se o open source software využívající primárně skriptů psaných v jazycích Python a XML,

schopný uživatelsky přívětivějšího nastavování jednotlivých dopravních pravidel. Obsahuje rovněž grafické rozhraní, na kterém lze jednotlivé simulace pozorovat.

Práce si dává za cíl podobně prozkoumat trendy, které se uplatňují v již instalovaných systémech světelných křižovatek. Následně vytvořit prostředí pro simulaci zmiňovaných systémů. V neposlední řadě navázat na problematiku optimálního nastavení těchto dopravních uzlů, aby v maximální možné míře docházelo k potlačení či alespoň zmírnění důsledků asymetrie dopravní zátěže. Díky parametrům jakými jsou celkový čekací čas a nejdelší čekací čas lze vyhodnocovat schopnost algoritmů přiblížit se optimální délce zelených front.

2 PŘEHLED SOUČASNÉHO STAVU POZNÁNÍ

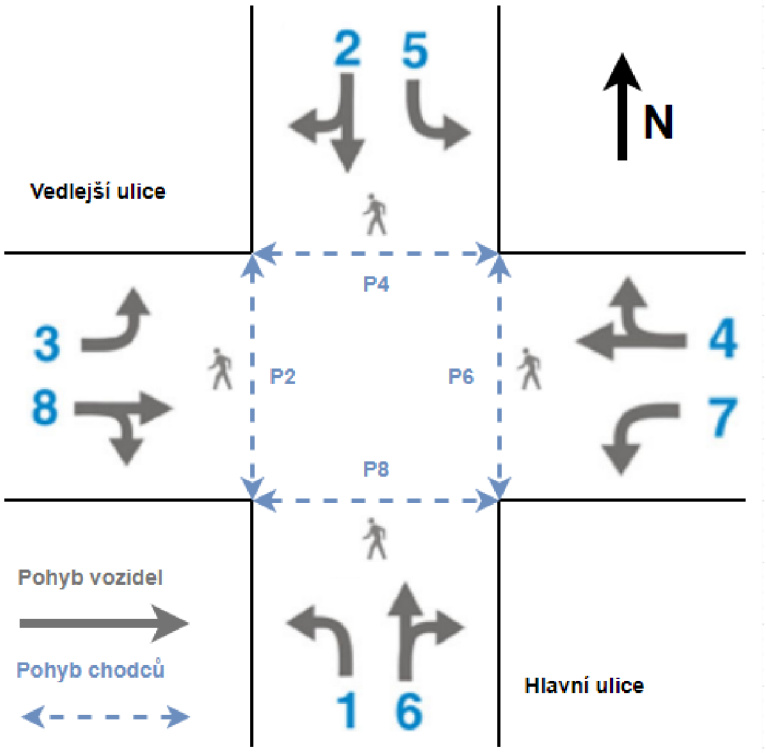
2.1 Definice základních pojmů

Protože se terminologie používaná ve výzkumných dokumentech ITSCP (Intersection Traffic Signal Control Problem) po desetiletí neustále mění, je vhodné začít jednoznačnou definicí dále užívaných pojmů. Řeč je o terminologii spojené s dopravou a dopravními uzly jako takovými.

Na křižovatce se pohyb různých účastníků provozu, jako jsou vozidla a chodci, řídí pravidly indikovanými dopravními signály. Tyto signály přichází ať už ve formě statistických dopravních pravidel, tak ve formě světelných křižovatek. V tradičním nastavení světelných křižovatek existuje sekvence indikací, které se periodicky opakují. V současnosti jsou preferovány tři hlavní pojmy popisující nastavení světelných křižovatek: cyklus, fáze a trvání.

Cyklus je celkový čas potřebný k dokončení jedné signalizační sekvence pro všechny pohyby na křižovatce. Fáze je doba, po kterou je spuštěn jeden nebo více pohybů formou zelené indikace. Trvání je doba, po kterou signál stráví v každé fázi, během níž se indikace nemění.

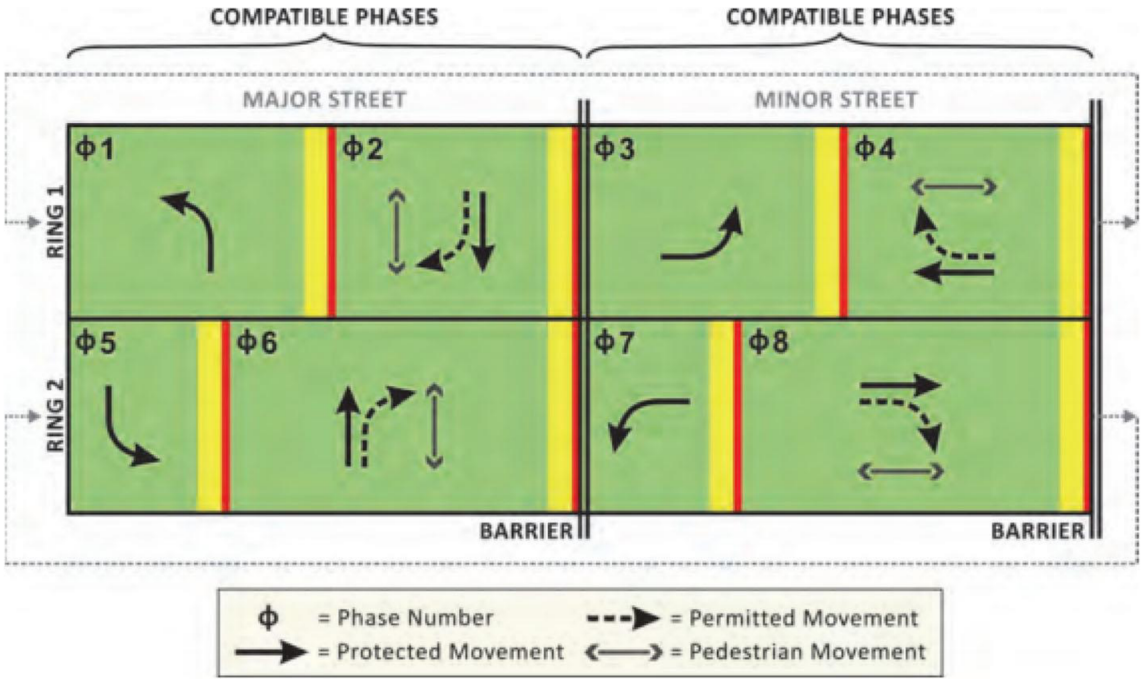
Dále je skupina dopravních toků definována jako jeden nebo více kompatibilních pohybů účastníků silničního provozu a každá fáze má sadu časování pro každou skupinu dopravních toků. Obrázek 1 znázorňuje osm fází typických pohybů vozidel a chodců na křižovatce se čtyřmi směry, ve kterých plné a přerušované čáry představují pohyby vozidla a chodce. Každé číslo na obr. 1 odpovídá fázi; například vozidla směřující na západ a odbočující doprava a chodci směřující na západ a východ, kteří překračují severní úsek křižovatky, jsou zařazeni do fáze 4 [1].



Obr. 1: Typické uspořádání dopravních pruhů a pohybu chodců v rámci světelné křižovatky [1].

Kombinací pojmů cyklus, fáze a trvání lze definovat sekvenci fáze signálu, která představuje sled pohybů vozidel ve zkoumaném dopravním uzlu. Obrázek 2 znázorňuje typický příklad sledu fází signálu na čtyřramenné křižovatce, nazývané standardní kruhový a bariérový diagram National Electrical Manufacturers Association (NEMA).

V tomto diagramu je prstenec posloupností fází, které jsou nekompatibilní, a proto musí být obsluhovány v určitém pořadí. Bariéra je referenčním bodem v cyklu, ve kterém fáze v každém prstenci dosáhla svého bodu ukončení. Na **obr. 2** musí být fáze v obou prstencích na závoře současně zbarveny do červena. K definování sledu fází signálu lze použít různé kombinace a pořadí fází, pokud je zaručeno zabránění konfliktních pohybů. Těmito pohyby se rozumí situace, ve kterých by teoreticky došlo k obousměrné přednosti účastníků silničního provozu. Příkladem takové situace může být rozsvícení vyklizovací šipky spojená zároveň se zelenou z opačného směru. Nemyslí se však situace, ve kterých je hierarchie přednosti jasně definována [1].



Obr.2: Typický příklad sekvence fáze signálu (NEMA kruhový a bariérový diagram) [1]

2.2 Úloha optimalizace

Základní optimalizační problém je následující [2]:

$$\min_{x} \text{imalizace } f(x)$$
$$\text{za podmínky } x \in X \tag{1}$$

proměnná x je v tomto kontextu design point, neboli bod návrhu. Ten může být reprezentován jako vektor hodnot odpovídajících různým proměnným modelu. Dále tedy n-rozměrný bod návrhu lze napsat jako:

$$[x_1, x_2, \ldots, x_n] \tag{2}$$

kde i-tá proměnná modelu je označena jako x_i. Prvky vektoru (2) je žádoucí volit tak, aby došlo k minimalizaci účelové funkce f. Jakákoli hodnota x ze všech bodů v definičním oboru X, která minimalizuje účelovou funkci, se nazývá řešení nebo také minimalizátor. Konkrétní řešení se zapisuje notací x^*.

Tato formulace je obecná, což znamená, že být jakýkoli optimalizační problém přepsán podle rovnice (1). Zejména problém:

$$\underset{x}{\text{maximalizace}}\ f(x) \text{ za podmínky } x \in X \tag{3}$$

lze nahradit problémem:

$$\underset{x}{\text{minimalizace}}\ -f(x) \text{ za podmínky } x \in X \tag{4}$$

Nový tvar je pouze jiný syntaktický zápis původního problému, protože má stejnou množinu řešení. Modelování inženýrských úloh v rámci této matematické formulace může být náročné. Způsob, jakým jsou optimalizační problémy formulovány, totiž výrazně ovlivňuje složitost. Dále následuje zaměření se na algoritmické aspekty optimalizace, které vyvstávají po vhodné formulaci problému.

Tato práce dále pojednává o široké škále různých optimalizačních algoritmů. Obecně neexistuje žádný důvod, proč upřednostňovat jeden algoritmus před druhým, jestliže nepředpokládáme určité pravděpodobnostní rozdělení v prostoru možných účelových funkcí. Pokud jeden algoritmus funguje lépe než jiný algoritmus v jedné třídě problémů, pak bude v jiné třídě problémů fungovat hůře. Podmínkou mnoha optimalizačních algoritmů k efektivnímu fungování je taková, že je třeba, aby v cílové funkci existovala určitá pravidelnost. Příliš mnoho náhodnosti tak dokáže otřást jak se schopností konvergovat, tak i s validitou výsledků [2].

2.3 Volba simulačního software

Analytické modely jsou velmi užitečné, pokud jde o poskytování vhledů do obecnějšího chování systému. Simulační nástroje mají však prominentní místo při analýze reakcí dopravních systémů za proměnlivých podmínek. Simulace je velmi užitečný nástroj v dopravním inženýrství, kde se používá k analýze komplexního dynamického chování dopravních toků. Simulaci lze definovat jako imitaci reálných systémů, či přiblížení se k nim, nebo procesů pro pohodlné získávání informací prostřednictvím analogických modelů dopravních toků.

Tyto modely pomáhají při analýze reálného toku dopravy. Využití dopravních simulačních modelů je klíčové pro komplexní průzkum městského dopravního systému v bezpečném a vhodném prostředí. Jako celek lze dopravní simulaci široce dichotomizovat na mikroskopické a makroskopické přístupy. Ziskem je logické členění a vyšší přehlednost.

Přístup mikroskopické simulace uvažuje individuální chování řidiče spolu s interakcí s ostatními vozidly nebo chodci, zatímco makroskopický přístup bere v úvahu proudění vozidel jako celek. Mezoskopický je další přístup, který je hybridem výše uvedených přístupů. AIMSUN, CORSIM, MATSim, Paramics, SUMO, VISSIM jsou

některé z široce používaných mikrosimulačních balíčků pro zkoumání široké škály dynamiky [3].

Dále proběhlo vyhodnocení vlastnosti a charakteristiky různých běžně používaných balíčků dopravní simulace. Byla taky vytvořena relativní analýza se zaměřením na některé speciální funkce. Výstupem této práce bylo, že VISSIM, AIMSUN a CORSIM jsou vhodné pro modelování prvků, jako jsou dopravní tepny, kongesce na dálnicích a integrovaná síť dálnic a ulic, a AIMSUN, CORSIM a PARAMICS pro inteligentní dopravní systémy (IDS) [4].

Na základě analyzovaných dat bylo roztřízeno a porovnáno sedm nejpoužívanějších mikrosimulačních nástrojů na základě jedenácti funkcí, výsledky jsou uvedeny v tabulce 1. Pokud jde o grafické uživatelské rozhraní (GUI), všechny výše uvedené balíčky jsou shledány stejně uživatelsky přívětivými a adekvátními. Pro uzavírky jízdních pruhů a modelování pracovních zón má TransModeler velké výhody oproti AIMSUN a VISSIM.

Pokud jde o modelování rozhodování o směrování vozidel, VISSIM umožňuje snadné zadávání velkého množství dat prostřednictvím tabulek Excel. AIMSUN, VISSIM a SUMO jsou simulační nástroje, které umožňují uživateli vytvářet a ovládat modely pomocí programování aplikací.

Rozhraní (API) externím programovacím jazykem. „AIMSUN Next“ od AIMSUN poskytuje sadu nástrojů pro automatizaci úloh prostřednictvím programovacího prostředí. Component Object Model (COM) a Traffic Control Interface (TraCI) jsou programovací rozhraní poskytovaná VISSIM a SUMO. V obou případech se jedná o robustní struktury vytvořené formou knihoven. O knihovně TraCI je pojednáváno dále v práci. Dostupnost hierarchického objektového modelu VISSIM-COM usnadňuje kódování sítě pomocí VISSIM ve srovnání s SUMO. AIMSUN dříve dokonce neměl žádnou API. Nejdůležitější devízou je však jeho rychlost, která převyšuje všechny ostatní současně používané mikroskopické simulační nástroje.

AIMSUN se pokusil překonat všechny své nedostatky prostřednictvím AIMSUN Next. Kromě AIMSUN Next je doporučováno, aby tvůrce modelu měl střední znalosti o programování v Pythonu, C++ nebo VBA, pokud používá VISSIM a SUMO pro modelování. V opačném případě by vhodnou volbou mohl být AIMSUN. Kromě MatSim poskytují všechny výše uvedené mikrosimulační nástroje možnosti pro 2D i 3D vizualizaci [3].

Tab. 1: Klasifikace nejběžněji používaných software pro mikrosimulace [3].

	CORSIM [5]	MATSim [6]	AIMSUN [7]	Paramics [8]	VISSIM [9]	SUMO [10]	TRANSIMS [11]
Open Source	NE	ANO	NE	NE	NE	ANO	ANO
Systém	Diskrétní	Spojitý	Spojitý	Diskrétní	Spojitý	Spojitý	Diskrétní
Vizualizace	2D/3D	2D	2D/3D	2D/3D	2D/3D	2D/3D	2D/3D
Chodci	ANO	NE	ANO	ANO	ANO	ANO	NE
Busy/Tramvaje	NE	NE	ANO	ANO	ANO	ANO	NE
Oblast působnosti	Město/Region	Město/Region	Město/Stát	Město/Region	Město/Region	Město/Region	Město/Stát
Výstup	XML, CSV	Text	Grafy	HTML, XML, CSV	XML	XML	XML
Import map	/	ANO	ANO	/	ANO	ANO	/
Progmaovací jazyk	/	/	Python, C++	/	C++, Matlab, VB, Python,…	C++, Matlab, VB, Python,…	/
Flexibilita v rozvoji infrastruktury	Limitovaná	Limitovaná	Flexibilní	Flexibilní	Flexibilní	Flexibilní	Limitovaná
Programování	/	/	Složité	/	Jednoduché	Složité	/

Vzhledem k analýze výše zmíněných simulačních software, které se aktuálně nejhojněji používají k vytváření modelů dopravních toků a jejich simulaci, bylo zvoleno prostředí SUMO, tedy Simulation of Urban Mobility od společnosti Eclipse. Hlavními důvody pro tuto volbu byly vysoká míra modularity i přístup k licencování svého produktu. Dále je nutné zmínit vysokou míru flexibility, kterou daný software poskytuje. Možnost přímo importovat dopravní data z mapy a tím tedy vytvořit přesné dopravní situace. Dále se jedná o software, který je má velmi dobrou uživatelskou podporu na fórech či například na Githubu.

2.4 Kategorizace metodik pro nastavování světelného značení

Řízení dopravních signálů je jednou z nejúčinnějších metod řízení městské dopravy, která poskytuje plynulejší a bezpečnější dopravní tok na každé křižovatce. Otvírá se tím totiž možnost adaptivního řízení ve smyslu přizpůsobení se trendům dopravy v jednotlivých časech. Od doby, kdy byl zaveden jednoduchý automatický signálový regulátor, prochází tento systém nekončícími vylepšeními. Některé z překážek aplikace jednotlivých postupů jsou nedostatečná dopravní infrastruktura, rostoucí počet vozidel, povětrnostní podmínky, struktura dopravní sítě, výzvy spojené s průjezdy vozidel s předností jízdy atd.

Každá z těchto překážek je významná sama o sobě a má plný potenciál kdykoli generovat přetížení v síti. Některé formou dočasného ucpání, jiné jako fenomén komplikují dopravu téměř neustále. Zmírnění dopravních zácp způsobených těmito důvody je tedy náročným, komplexním a nelineárním stochastickým problémem, který musí inženýři a výzkumníci řešit [12].

V následujících odstavcích jsou zmíněny algoritmy a postupy řízení dopravních signálů. Jedná se o postupy, algoritmy a rodiny optimalizačních balíčků využívající

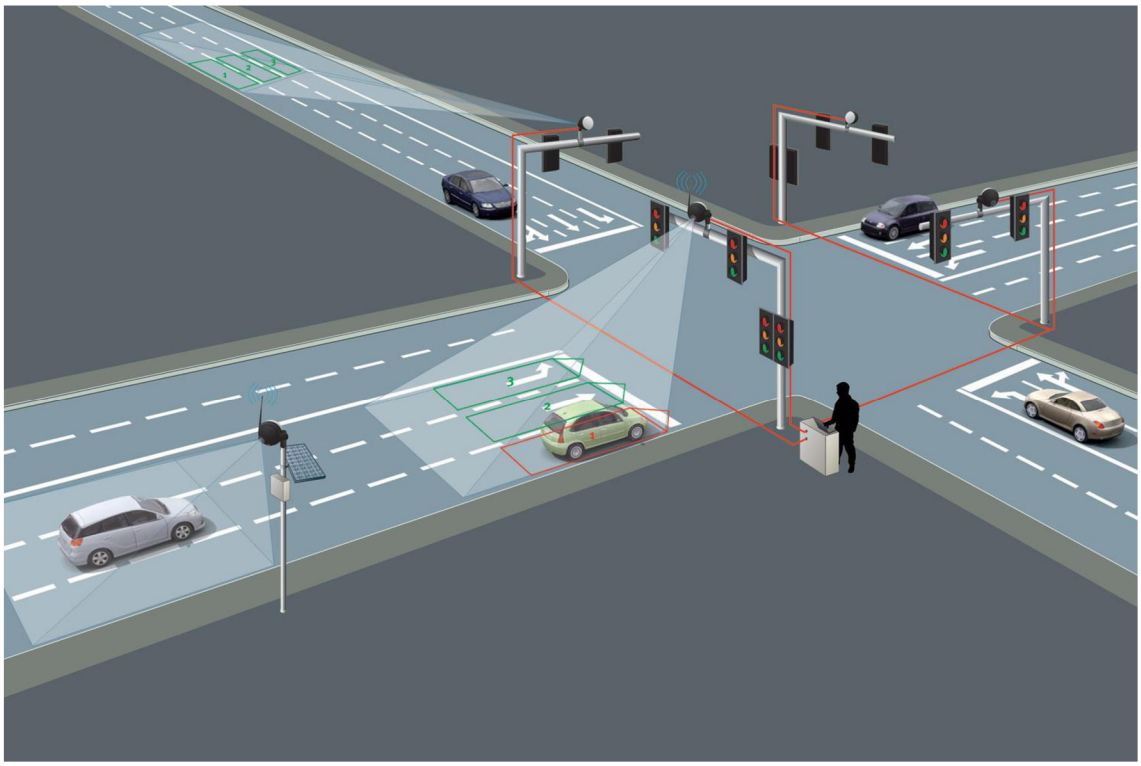
modely microSimOpt (Microsimulation-Based Optimization). Každý z těchto postupů skýtá výhody, vždy typicky spojeny se zvýšenou výpočetní náročností a robustností daného řídícího systému.

2.4.1 Model založený na mikrosimulaci

SimOpt, tedy simulační optimalizace, je obecně považována za obor, ve kterém jsou optimalizační techniky integrovány se simulační analýzou. Důvodem je najít nejlepší hodnoty rozhodovacích proměnných mezi všemi možnostmi bez explicitního vyhodnocení (simulování) každé možnosti. Simulační optimalizace pro nastavování délek světelných fází je důležitá, protože vyhodnocení účinků menších změn v rozhodovacích proměnných týkajících se světelných front lze přesně posoudit pomocí mikrosimulace bez skutečné implementace.

Pro pochopení konceptu; navrhovaná strategie či algoritmus žádá mikrosimulační model sítě, aby vyhodnotil aktuální řešení a výsledky. Z každého běhu simulace jsou výsledky přiváděny zpět do navrženého algoritmu, dokud není splněno nějaké kritérium ukončení. Používány jsou nesčetné přístupy a metody spolu s nástroji simulace dopravy v oblasti délek světelných fází. Tyto přístupy zahrnují přístupy strojového učení, fuzzy logiku a výpočetní strategie, jakými jsou Evoluční Výpočty (EV), Hejnová inteligence (HI) a další populační metaheuristické algoritmy [3].

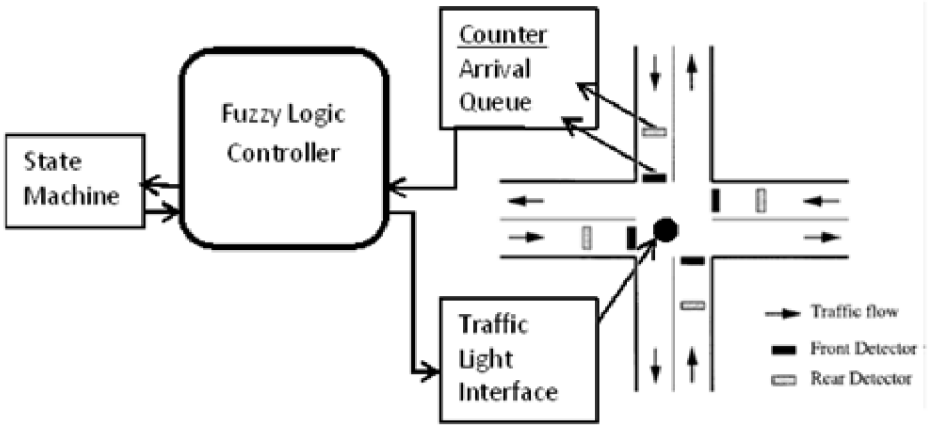
Schopnost učit se ze zkušeností je jednou z charakteristik metod umělé inteligence, díky které jsou tyto metody vhodné k řešení problémů reálného světa. Metody samoorganizující se umělé inteligence jsou odolné vůči dynamickým změnám podmínek. Navíc dochází k částečnému překonání nadměrné výpočetní náročnosti díky využití distribuovaných řídicích systémů. V distribuovaných modelech může mít každá křižovatka svůj vlastní ovládací prvek s vlastním zatížením dopravních dat. Distribuované regulátory jsou schopny zahrnout všechny výhody multiagentního systému, které zvyšují spolehlivost, robustnost a přesnost systému. Jedná se o řešení silně preferované v případě simulace či optimalizace velké dopravní sítě.



Obr. 3: Zjednodušené vykreslení adaptivní logiky semaforové křižovatky [13].

Další možností je využití Fuzzy logiky, tedy FLS (Fuzzy Logic System). FLS je metoda vhodná k vyjádření neurčitosti lingvistických frází. Dokáže tedy místo binární reprezentace dat pracovat s mírou splnění nějakých podmínek či vlastností. Ve skutečnosti je možné zpracovávat nepřesná data a nejisté informace pomocí fuzzy množin. Použití fuzzy teorie místo ostré teorie množin poskytuje možnost implementovat scénáře reálného světa podrobněji. Jednou z nejdůležitějších vlastností FLS je schopnost zahrnout do jejich návrhu znalosti odborníka. Navíc jsou transparentní, díky čemuž jsou pro operátory srozumitelnější ve srovnání s modely black box NN (Neural Network). Výstup těchto sítí je tedy jasně predikovatelný, jelikož lze každé ohodnocení či rozhodnutí zpětně dohledat. U neuronových sítí není podobná věc možná.

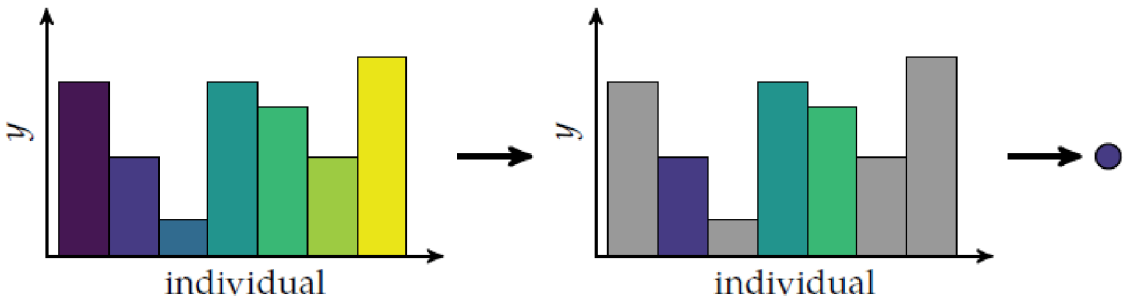
FLS mapuje vstupy na výstup systému. V situaci, kdy definice shluku nebo třídy objektů není nejasná, existuje pouze jednoduchá dvouhodnotová charakteristická funkce, nula a jedna. Fuzzy množinou je tato doména rozšířena na rozsah celých čísel mezi nulou a jedničkou. Pro reprezentaci některých lingvistických hodnot je doména systému rozdělena do fuzzy množin, například můžeme definovat nízký, střední a vysoký provoz. Poté se použijí funkce příslušnosti k zobrazení stupně závislosti na každé fuzzy množině. Každá vstupní hodnota může patřit do více než jedné fuzzy množiny. Podobnou situaci lze uvažovat o výstupním prostoru. Přidružování číselných hodnot k fuzzy množinám je fuzzifikace, defuzzifikace je název opačného procesu. Logika systému je definována pravidly if-then ve fuzzy inferenci [14].



Obr. 4: Schéma fuzzy logiky využité k ovládání délky zelené fronty světelné křižovatky [15].

Genetický algoritmus (GA), jak bylo teoreticky i empiricky dokázáno, nabízí konzistentní výkon v komplexních vyhledávacích prostorech. Filozofie GA vycházejí z přirozených genetických procesů. Zahrnuje pojmy jako gen, chromozomy, potomstvo, generace, křížení a mutace.

GA začíná náhodně generovanou populací potomků jako počátečním řešením. Jedná se o ekvivalent k pojmu počet agentů u jiných přístupů. Ke zjištění optimální hodnoty vstupního parametru daného problému se využívá fitness nebo hodnotící funkce. Algoritmus postupuje výběrem některých potomků podle určitých kritérií výběru a odmítnutím zbývajících řešení. Následně GA vytváří novou generaci potomstva pomocí reprodukčního procesu. Tento proces se skládá ze tří částí: selekce, křížení a mutace. Při selekce dochází k výběru potomků na základě hodnoty jejich účelových funkcí. Možných algoritmů pro výběr je celá řada, od obyčejného ruletového kola přes turnajový výběr až k elitnímu turnajovému výběru. Křížení je postup, při kterém dochází k částečnému překrytí dvou vybraných jedinců a vytvoření tak jedinců nových. Mutace je pak doplňkovou funkcí všech genetických algoritmů, která pro velmi malé množství mění jejich bitovou či jinou reprezentaci a tím i jejich hodnotu. Mutace konkrétně má největší vliv na schopnost dostávání se z lokálních extrému. Zároveň však snižuje míru a rychlost konvergence. Všechny tyto složky společně rekombinují a vyvíjí novou generaci. V průběhu optimalizace se iteruje celá řada generací, dokud algoritmus nedojde k nejlepšímu řešení nebo nedosáhne kritéria zastavení určeného pro daný problém [16].

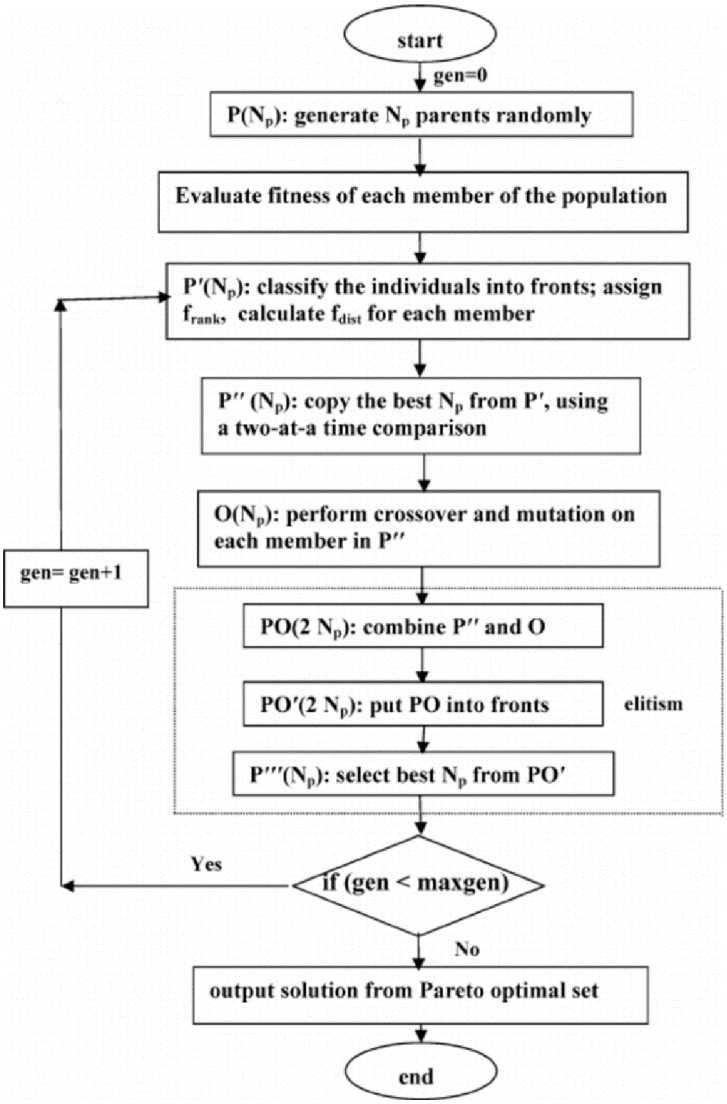


Obr. 5: Ukázka turnajové selekce genetického algoritmu [2].

Na obrázku výše se nachází příklad jednoho z typů selekce. Při užití turnajového výběru je každý rodič, tedy předloha pro vytváření chromozomů nové generace, vybírán jako nejslibnější z náhodně zvolených chromozomů, neboli jedinců populace. Konkrétně je vyobrazen turnajový výběr s velikostí populace m = 7 a velikostí vzorku k = 3. Tento proces se provádí pro každého rodiče zvlášť. Výška sloupce označuje hodnotu jeho fitness funkce, zatímco jeho barva označuje o jakého jedince se jedná. Úkolem výše byla minimalizace, dochází tak k lepšímu ohodnocení pro nižší hodnoty y. Ze tří zvolených jedinců v pravé části obrázku tak vyhrává tmavě modrý. Tento jedinec se zapíše dále a proces je opakován. Jedinci, kteří neprojdou touto selekcí, jsou odstranění z řešení a v případě softwarové interpretace uvolněni z paměti [2].

Multiobjektivní Evoluční Algoritmy (MOEA) a dopravní simulátory se široce používají k optimalizaci časování dopravních signálů s více sledovanými parametry. Simulace provozu však vyžadují mnoho času na zpracování a je třeba je volat opakovaně v iteracích MOEA. V důsledku toho je proces optimalizace časování dopravního signálu výpočetně náročný, kdykoli je stavový prostor příliš velký či omezovací podmínky příliš slabé. Proto je v algoritmech optimalizace časování dopravních signálů žádoucí sledovat chování jako celek [17].

Jednou z hlavních výhod evolučních, případně genetických, algoritmů spočívá v rozumné rovnováze mezi dvěma hlavními parametry optimalizačních algoritmů: exploration (průzkum) a exploitation (využívání). První zmíněný parametr nutí algoritmus neustále prohledávat nové oblasti stavového prostoru a tím zamezuje zacyklení uvnitř lokálního extrému. Hlavním nástrojem je zde proces zvaný mutace a částečně křížení. Druhý zmíněný parametr, využívání, popisuje míru snahy nalezení extrému uvnitř ohraničené oblasti. Hlavními nástroji jsou selekce a částečně křížení. Jedná se tedy o protichůdné tendence, neboť průzkum snižuje rychlost konvergence zatímco využívání ohraničuje prozkoumávaný stavový prostor a má tendence zacyklení. Správně vybalancovaný algoritmus je tak schopen efektivního přiblížení se lokálním extrémům. Vývojový diagram takového algoritmu se nachází na obrázku níže.



Obr. 6: Vývojový diagram multiobjektivního evolučního algoritmu (MOEA) [18].

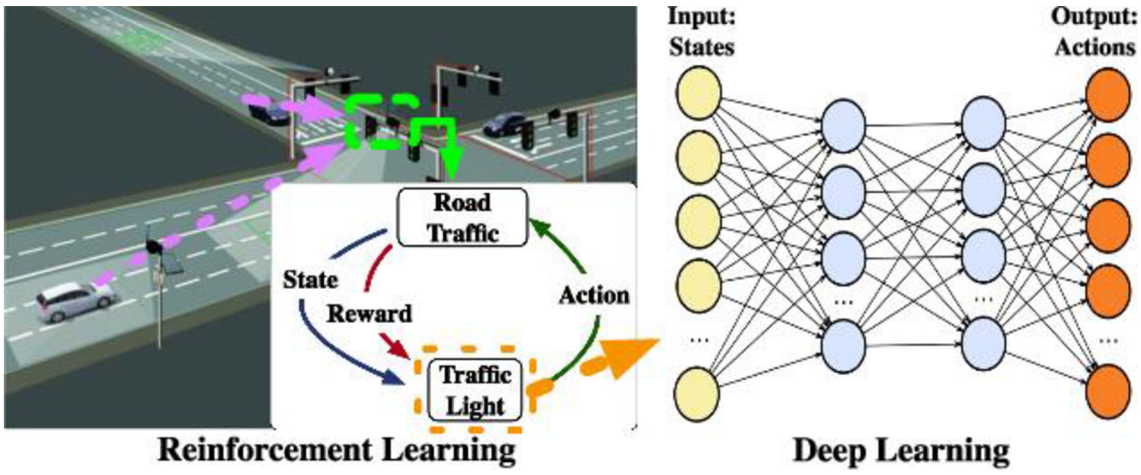
2.4.2 Modely založené na výpočetní inteligenci

Modely v této části používají různé funkce odhadu k vyhodnocení potenciálních řešení během procesu vyhledávání. To proto, aby došlo k nalezení optimální nebo téměř optimální řešení v co nejkratším čase. Některé metody či algoritmy také v menší míře využívají mikrosimulační nástroje. Využití těchto nástrojů zde však neslouží k nalezení řešení, nýbrž spíše k demonstraci potenciálních přínosů navrhovaného řešení, tedy k vyhodnocování ve formátu uživatelsky přívětivým. V této kategorii jsou diskutovány přístupy jako fuzzy modely, neuronové sítě, algoritmy strojového učení, evoluční výpočty (EV), Hejnová Inteligence (HI) a další populační metaheuristické algoritmy.

Inspirace pro EV i HI algoritmy často pochází z přírody, příkladem je biologická evoluce. Klasické EV zahrnují evoluční strategie, evoluční programování, GA a genetické programování (GP). Všechny zmíněné rodiny algoritmů jsou metaheuristickými optimalizačními technikami pro nalezení optimálního nebo téměř

optimálního řešení nelineárního komplexního problému v přijatelném časovém limitu. Napodobují přirozené procesy, jakými jsou přirozená evoluce podle principu fit nebo přizpůsobení nejlepším podmínkám prostředí, dobře známým jako fenomén přežití nejschopnějších. Algoritmy SI mají původ v chování některých společenských živých bytostí, jako jsou ptáci, mravenci, včely či například termiti. Hlavní síla výzkumu založeného na SI závisí hlavně na dvou rodinách algoritmů, konkrétně na optimalizaci mravenčích kolonií a optimalizaci hejna částic. Oba tyto přístupy jsou velmi úspěšné v různých druzích optimalizačních problémů.

Posilované učení je podoblastí strojového učení. Charakterizováno je maximalizací numerické odměny a následným mapování stavu do akcí, a to různými způsoby. Jedná se o jedno ze základních tří paradigmat strojového učení, spolu s učením s učitelem a učení bez učitelem. Samotné prohledávání stavového prostoru za účelem nalézání optimálního řešení probíhá za pomoci agentů, kteří interagují s prostředím. Prozkoumání metodou pokusu a omylu je nejdůležitější vlastností agenta. Agent, který přetrvává příliš dlouho v nějaké situaci, se může poučit ze zkoušek, aby si tak vybral akci, která poskytuje největší odměnu. Řešení je následně vybíráno jako pozice či účelová funkce jednoho z agentů [19].



Obr. 7: Využití posilovaného a hlubokého učení u řízení světelných semaforů [20].

Algoritmus umělého včelstva (ABC) je příkladem populační metaheuristiky. Hlavní inspirací je potravní chováním včelstva. V ABC existují tři druhy včel, a to zaměstnané, přihlížející a skautské. Každé řešení uvažovaného problému se nazývá zdroj potravy, přičemž vhodnost řešení odpovídá množství nektaru souvisejícího zdroje potravy. Základními kroky algoritmu jsou inicializace, po které následuje iterační smyčka každé skupiny včel postupně. Algoritmus běží dokud není splněno ukončovací kritérium, ať už přiblížení se optimu či dosažení časového limitu [21].

Tradiční metodou optimalizace problému s mnoha parametry je vyhledávání v mřížce, takzvaný Grid search. Ten spočívá v kompletním vyhledávání v dané podmnožině prostoru parametrů trénovacího algoritmu. Protože prostor parametrů algoritmu strojového učení může obsahovat mezery se skutečnými nebo neomezenými hodnotami pro některé parametry, je možné, že bude nutné určit hranici pro použití vyhledávání v mřížce. Vyhledávání v mřížce trpí vysokou výpočetní náročností pro vysoce rozměrové prostory. Často je lze snadno rozjíždět paralelně, protože hodnoty parametrů, se kterými algoritmus pracuje, jsou obvykle na sobě nezávislé [22].

3 VLASTNÍ ŘEŠENÍ

3.1 SUMO software

Jedná se o open source, multifunkční balíček nástrojů pro simulaci dopravy, který dokáže reprodukovat realistické modely dopravních toků v simulačních scénářích. Poskytuje vysokou flexibilitu pro modelování konvenčního i automatizovaného řízení v jakémkoli dopravním scénáři. Dále je vhodný pro další zkoumání kontrolních opatření v libovolném dopravním scénáři [23].

Při implementaci řešení řízení dopravy je nezbytná přesná znalost dopravních podmínek a dynamiky. Přesněji řečeno je tato znalost velmi vhodná pro vytváření reprezentativních modelů. Rámce pro simulaci provozu poskytují užitečný nástroj pro zodpovězení složitých výzkumných otázek, pro hodnocení nebo testování strategií řízení dopravy a jejich dopadů. Nástroje pro simulaci dopravy lze rozdělit především do čtyř různých skupin:

- Makroskopické: simuluje se průměrná dynamika vozidla, jako je hustota provozu
- Mikroskopické: každé vozidlo a jeho dynamika jsou modelovány individuálně
- Mezoskopický: směs makroskopického a mikroskopického modelu
- Submikroskopické: každé vozidlo a také funkce uvnitř vozidla jsou explicitně simulovány, např. řadicí páka

Výhodou makroskopických modelů je obvykle jejich nízká výpočetní náročnost. Detailní simulace mikroskopických nebo submikroskopických modelů jsou však přesnější, zejména pokud by mělo dojít k simulaci emisí nebo jednotlivých tras, a to v řádu jednotek.

Aby bylo možné simulovat reálný provoz, je nezbytná řada vstupních parametrů. Nejdůležitějšími jsou parametry sítě (např. silnice a chodníky), další dopravní infrastruktura (např. semafory) a dopravní poptávka, tedy dopravní trendy a jejich objemy.

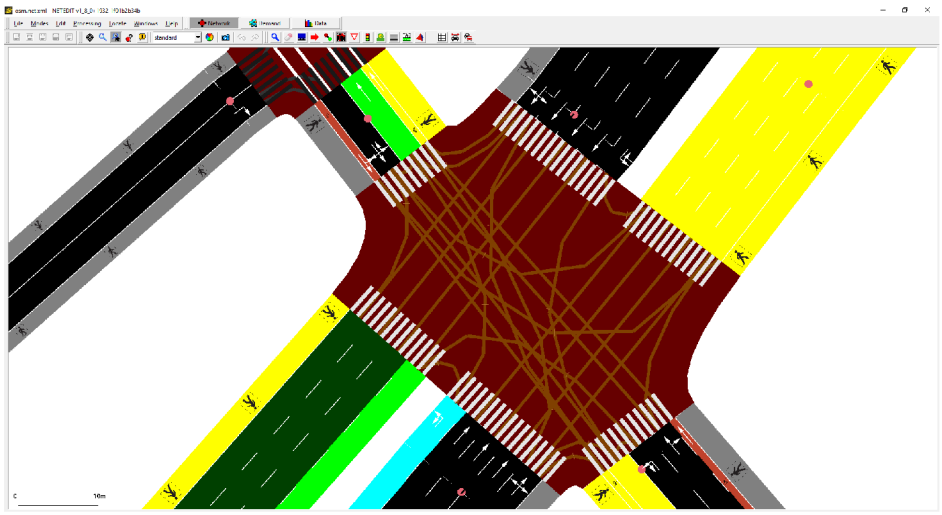
Společně tyto prvky tvoří simulační scénář. Vzhledem k tomu, že modely simulace provozu se obvykle používají pro stochastické chování, má smysl takový scénář simulovat několikrát a vyvodit statistické závěry. Statisticky významné soubory dat lze následně využívat v analýze výsledků, kde se na jejich základě určuje optimální nastavení dopravních uzlů.

Po definování scénáře může být užitečné pozorovat objekty simulace (vozidla, chodce, semafory) ve vizuální reprezentaci pro kvalitativní ověření. Typicky je tato možnost výhodná při manuální změně definice modelu přímo v XML souborech, ve kterých jsou data uložena. Za tímto účelem poskytuje SUMO aplikaci SUMO-GUI, která umožňuje pozorování simulace při různých rychlostech. Například také s různými možnostmi barevnými schématy pro zvýraznění různých aspektů, jakými jsou rychlosti, hustota provozu, převýšení silnice nebo v neposlední řadě pravidla přednosti v jízdě [23].

Aby bylo možné kvantitativně vyhodnotit simulaci, je poskytována široká škála výstupních souborů, které lze selektivně aktivovat, jako například:

- Trajektorie vozidel (polohy a rychlosti)
- Dopravní data shromážděná z modelovaných detektorů
- Údaje o provozu agregované přes síťové prvky (hrany nebo jízdní pruhy)

Sítě SUMO se skládají z uzlů a jednosměrných hran představujících ulice, vodní cesty, tratě, cyklostezky a chodníky. Seznam podporovaných hran je odráží množství druhů dopravy, která je rovněž podporována. Každá hrana má geometrii popsanou řadou liniových segmentů a skládá se z jednoho nebo více pruhů probíhajících paralelně. Atributy jako šířka, rychlostní limit a přístupová oprávnění (např. pouze autobus) jsou modelovány jako konstantní podél jízdního pruhu. V důsledku toho musí být úsek silnice modelován jako sekvence hran, když se kterýkoli z těchto atributů mění podél jeho délky. K tomuto dělení je v nástroji pro editace vytvořena funkce Split, která je schopná hranu v libovolné délce rozetnout na dvě.



Obr.8: Šesticestná křižovatka rozebraná pomocí software SUMO [24].

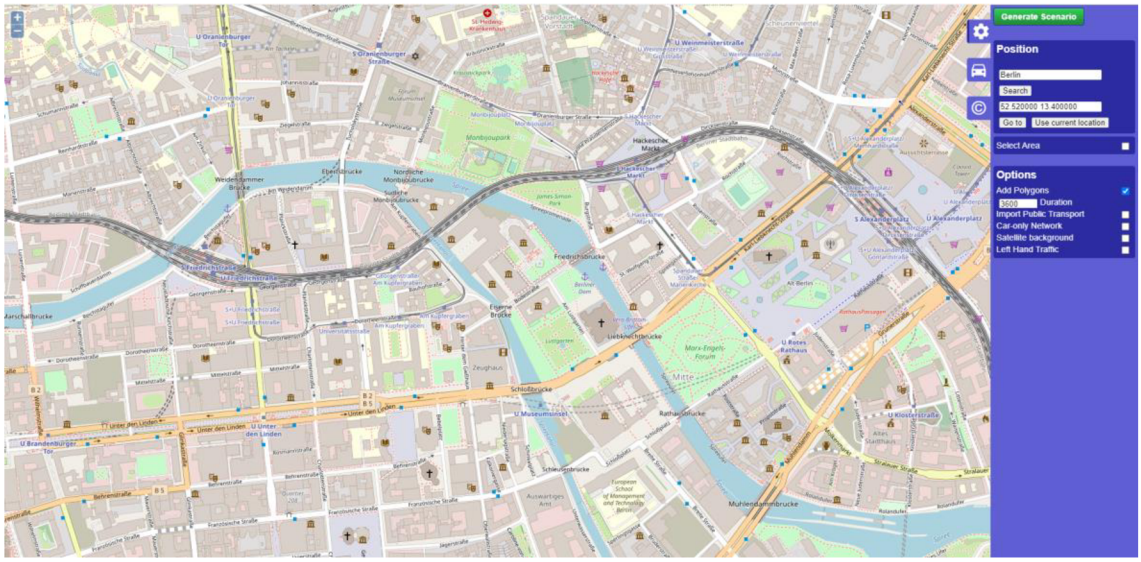
Obrázek 8 značí analýzu jednoho z dopravních uzlů ve složité síti městské dopravy. Jednotlivé hnědé křivky znázorňují možný pohyb účastníků provozu v jednotlivých pruzích. Žlutou jsou zvýrazněny pěší zóny a ostře červená je vyhrazena pro kola. Cihlově červená pak vyznačuje oblast samotné křižovatky, v níž se auta pohybují v závislosti na světelném značení.

Bílé zebrování znázorňuje přechody pro chodce. Světle zeleně zbarvené hrany informují o aktuální selekci. Pokud by byly prováděny nějaké změny, týkaly by se právě dvou hran nacházejících se na levé straně schématu.

Sítě SUMO obsahují podrobné informace o možných pohybech na křižovatkách a odpovídajících pravidlech přednosti v jízdě, která se používají k určení chování při dynamické simulaci. Pro zajištění konzistentní reprezentace sítě jsou sítě SUMO vytvářeny pomocí aplikací Netconvert a Netedit. Je nutné připomenout, že legislativa jednotlivých zemí ohledně pohybu na silnici není importována uživatelem, jedná se tedy o nativní proměnné získávané přímo ať už podle lokality či jiných indicií.

Netconvert je nástroj příkazového řádku, který lze použít k importu silničních sítí z různých zdrojů dat, např. OpenStreetMap (OSM), Opendrive, Shapefile nebo z jiných simulátorů, jako je MATSim a Vissim. Klíčovou vlastností nástroje Netconvert je heuristické zpřesňování chybějících síťových dat pro dosažení potřebné úrovně detailů pro mikroskopickou simulaci (tj. syntetizace plánů semaforů, pravidel přednosti v jízdě a geometrií křižovatek pro sítě OSM). Dochází tak k vytvoření funkční sítě ve smyslu absence kolizí a jednoznačné definici jednotlivých hran i jejich přechodů. Všechny tyto informace jsou spolu s modelem uloženy do jednotlivých souborů ve formátu XML [23].

Alternativou k nástroji Netconvert je OSM web wizard, který je zobrazen níže. Výhodou tohoto software je možnost nalezení na mapě inkriminované oblasti a tím získání přesně té části dopravy, která je předmětem zkoumání. Navíc si lze povšimnout několika parametrů pro konverzi, která se mohou ukázat být velmi užitečnými.



Obr. 9: Ukázka GUI nástroje OSM (Open Street Map) web wizard.

Netedit je grafický síťový editor, který lze použít k vytváření, analýze a úpravě síťových souborů. To slouží k doplnění heuristiky generování sítě o ruční upřesnění a také podporuje definování další dopravní infrastruktury, kterou nebylo možné importovat pomocí Netconvert či OSM web wizard. Nástroj Netedit je nezbytný pro zmíněný postprocessing u importovaného modelu, jelikož je zpravidla třeba doupravit jej podle reálné situace. Databáze OSM jsou závislé na uživatelských příspěvcích, proto se může model, z důvodu nedostatku dat, vygenerovat chybně či v naprosto neodpovídajícím stavu.

Vzhledem k častému nesouladu mezi dostupnými vstupními daty a nezbytnou úrovní detailů pro mikroskopickou simulaci, je příprava sítě a infrastruktury často náročným úkolem. Z tohoto důvodu je Netconvert neustále vyvíjen, aby se zlepšila jeho heuristika a snížilo se tak požadované množství ručních úprav, přestože jsou stále často nezbytné.

Jedním z předpokladů pro simulaci dopravy obecně v dané síti je zajištění dopravní poptávky. V prostředí SUMO lze dopravní poptávku definovat jako jednotlivé cesty, toky nebo jako trasy. Základní informace by měly zahrnovat čas odjezdu, původ, cíl a způsob dopravy tam, kde nebude zaručena jednoznačnost druhu účastníka silničního provozu. Pokud je dopravní poptávka definována jako trasy, informace o původu a cíli nejsou povinné. Místo toho jsou vyžadovány další informace o trase, definované jako sekvence hran.

Jakmile je vygenerován požadavek na provoz, lze provést přiřazení provozu v SUMO, tj. Duarouter, Marouter nebo Oneshot, pro analýzu stavu provozu zkoumané sítě. Výše uvedené metody přiřazení se liší především principy přiřazení (User Equilibrium (UE), Stochastic User Equilibrium (SUE) nebo nejrychlejší trasa v daném čase odjezdu).

Matice původu a cíle (O-D), jako jeden z výstupů v tradičním čtyř krokovém modelu prognózování dopravy, je typickým způsobem, jak popsat poptávku po dopravě mezi danými zónami analýzy dopravy. Každá buňka matice O-D popisuje příslušnou dopravní poptávku jako množství druhu dopravy, tedy například osobní a nákladní automobily. Taková data nelze přímo použít v SUMO. Nástroj od2trips poskytuje způsob, jak převést takové matice O-D na jednotlivé cesty. Pro generování času odjezdu pro každé vozidlo/osobu lze zvolit buď jednotné nebo náhodné rozdělení v daném časovém období. V současné době lze zpracovávat pole O-D buď ve formátu Vissim nebo Visum.

SUMO již poskytuje rozsáhlý framework s užitečnými nástroji pro generování, ověřování a vyhodnocování velkých dopravních scénářů. Stále však existují další funkce a rozšíření, která jsou plánována pro vylepšení a rozšíření SUMO. Již dochází k podpoře řady funkcí souvisejících s intermodální železniční simulací, jako jsou jízdní řády, železniční přejezdy, železniční návěstidla a dynamické modely vlaků. Pro rozšíření možností železničních simulací se do budoucna plánuje řada rozšíření [23].

Obr.10: Logo simulačního software SUMO i s jeho typickým zeleným pozadím [24].

3.2 Volba křižovatky

Dále byla tedy zvolena nikoli simulace uměle vytvořené křižovatkové situace, nýbrž cesta řízení jedné konkrétní z reálného prostředí. Dochází tak k odstranění možných slabin celé práce, jakými můžou být nízká aplikovatelnost do reálného provozu, či příliš líbivá aproximace nějaké opravdové dopravní situace. Data jsou tedy získána přímo z OpenStreetMap (OSM), a to na základě výběru. Pro další účely analyzování bylo rozhodnuto, že není žádoucí vybírat větší plochu než tu, kterou by daný ovládací algoritmus využíval k rozhodování o řízení zmíněné křižovatky. Hovoříme tedy o několika desítkách metrů z každé strany spojnice vedoucí do křižovatky.

Co se týče lokality, která byla zvolena, došlo zde k vyhodnocení několika parametrů současně. Jednak bylo žádoucí vybrat takový dopravní uzel, ve kterém se výrazněji projeví asymetrie dopravní zátěže. Tedy stav, kdy špičky poptávkových toků jednotlivých spojnic křižovatky jsou odděleny v čase. Typickým příkladem je ranní zhuštění dopravy ve směru do města a odpolední zvýšená koncentrace směrem z města. Jednak se muselo jednat o dostatečně robustní křižovatku, tedy alespoň čtyři spojnice a alespoň dva dopravní pruhy v každém z nich.

Jedním z rozhodovacích parametrů byla rovněž osobní zkušenost. Dá se předpokládat, že znalost dané křižovatky není na škodu ve vztahu k nastavování její vnitřní logiky. Navíc je samotná simulace dopravy v této křižovatce o to komfortnější, neboť jsou zevrubně známy její převažující slabé stránky. Lokalita této křižovatky je vykreslena na obrázku 11. Snímek je pořízen ze serveru Mapy.cz.

Obr. 11: Lokalita vybrané světelné křižovatky [25].

Jak je ze snímku patrné, jedná se o čtyřsměrnou světelnou křižovatku. Nachází se na periferii měst Uherské Hradiště a Kunovice. Třída Vítězství je ulice která tyto územní celky propojuje. Severním směrem se tedy nachází město Uherské Hradiště, směrem na jih město Kunovice. Z východu se napojuje silnice E55, shodou okolností druhá nejdelší cesta v Evropě. Západním směrem se nachází dva supermarkety a několik málo obchodů.

3.3 Tvorba modelu

3.3.1 Inicializace pomocí OSM web wizard

Aby bylo možné nějaké z výše uvedených přístupů aplikovat a vyhodnocovat data z nich vycházející, je žádoucí vytvořit prostředí pro to určené. Jak již bylo zmíněno, součástí balíčku SUMO software je také OSM web wizard, tedy nástroj, který je schopen z výběru na mapě vytvořit detailní dopravní simulaci jemu odpovídající.

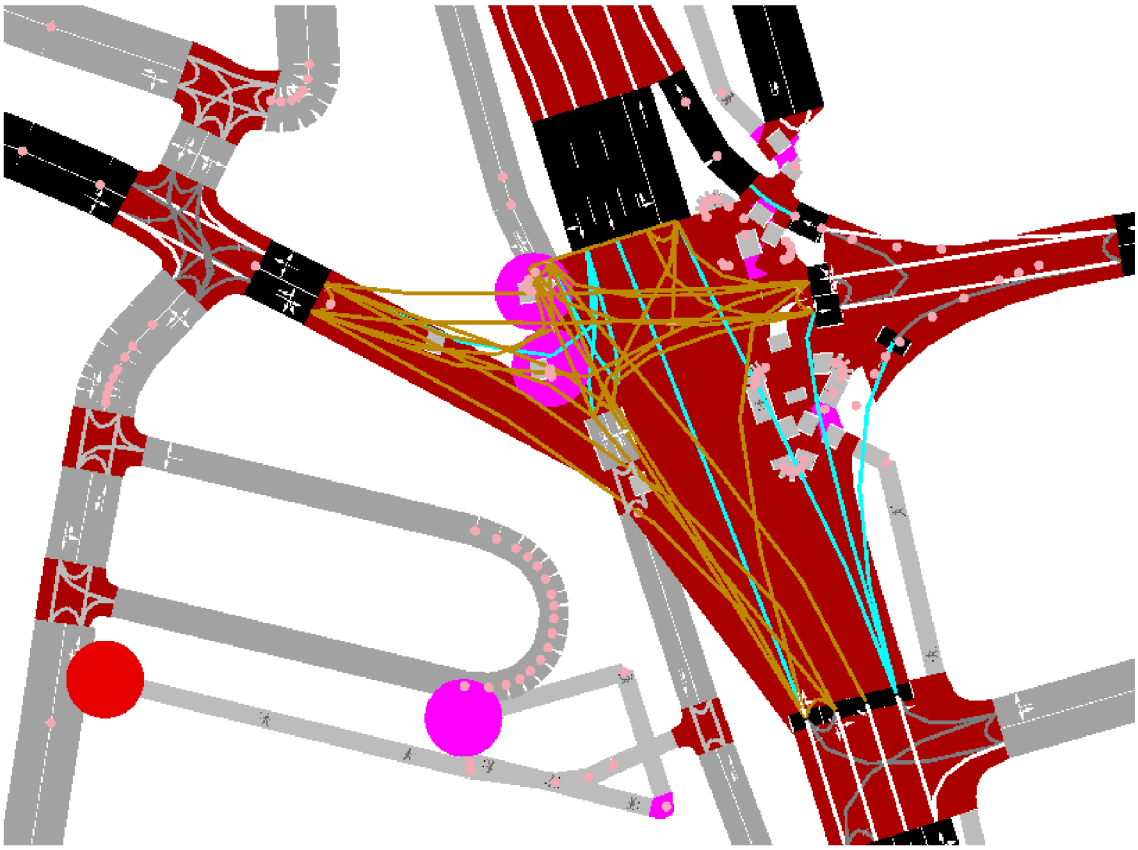
Obecně se tak nabízely dva možné přístupy. Jednak vytáhnout data z datové struktury Open Street Map (OSM), jejíž přesnost však závisí na dobrovolných příspěvcích dat od jejich spolutvůrců. Druhou možností pak je modelování jednotlivých dopravních spojů tzv. na zelené louce, tedy v naprosto prázdné simulaci. Vzhledem k již použitým argumentům a dalšímu rozmyšlení byla využita zásoba dat OSM. Za pomoci nástroje OSM web wizard a následných kompilátorů tak byl vytvořen následný model:

Obr. 12: Prvotní generování křižovatky za pomoci OSM web wizard.

Došlo tak k velmi násilné změně rázu nejenom krajiny ale i samotné logiky dopravního uzlu. Uvnitř černé plochy, která má signalizovat přítomnost silnice určené pro automobilovou dopravu, je značné množství děr. V tomto kroku se ukázalo, že nástroj používaný k transformaci dat z map do simulační podoby má velké mezery v oblasti vykreslování dopravních ostrůvků. Hned tři ostrůvky totiž nezobrazil vůbec, a z těch co zaznamenal, nezobrazil správně ani jeden.

Dalším úskalím bylo množství zobrazovaných komunikací. Světle šedá značí užitkové cesty, které se opět konvertovaly ve značné distorzi. Jelikož se buď jedná o pěší nebo o obslužné cesty, které nemění charakter ani ráz sledované dopravy, byly tyto dopravní uzly i sítě odstraněny. Toto tzv. trimování je velmi často nezbytnou součástí analýzy dopravní sítě skýtající vysoké desetitisíce dopravních spojnic. Dochází tak k odstranění redundantních spojnic typicky v hustě protkaných obytných zónách či parcích, kde zobrazování natolik detailní dopravní situace není výhodné [26].

V neposlední řadě je nutné všimnout si samotné jízdní logiky dopravního uzlu. Křižovatka jako taková je rozsekána do mnoha jednotlivých dopravních křížení, které na sebe jen zdánlivě navazují. Dále jsou na první pohled náhodně posunuty hrany křižovatek, samotné jízdní pruhy se objevují a zanikají. Aby došlo k získání přehledu o kvalitě konverze z hlediska dopravní logiky a správného využití jednotlivých pruhů, je přiložen následující obrázek.



Obr. 13: Zobrazení jednotlivých propojení a oblastí křižovatek v nástroji NETEDIT.

Nyní následuje popis situace na obr. 13. Puntíky značí další napojení, nikoli však křižovatkové. Objevují se buď u styku komunikace pro pěší s komunikací pro pěší (fialový puntík) nebo s komunikací pro automobily (červený puntík). Veškeré černé jízdní pruhy jsou opatřeny směrovkami určujícími možný další pohyb prostředků po nich cestujících. Jednotlivé červené plochy představují oblasti křižovatek. Síťová struktura rozprostírající se po všech červených oblastech značí jednotlivé propojení uvnitř křižovatky. Libovolný prostředek vstupující do červené oblasti z určitého pruhu smí přejet pouze do propojených pruhů. Žlutá barva propojení značí zvýšenou hustotu dopravních spojnic a tím i potenciálně nižší dopravní propustnost z důvodu vysoké hustoty. Tyrkysová zobrazuje aktuální stav semaforu, a to propustný směr. Krom očividného vedení mnoha žlutých i tyrkysových propojení mimo červenou oblast si lze rovněž všimnout chaotického propojování ve směru na jih i západ.

Takto simulovaná křižovatka neobsahuje žádné kolize, tedy překrývající se směry se zelenou frontou. Teoreticky je tak průjezdná a simulace se na ní s trochou odvahy rozjede. V žádném případě však ani zdánlivě neodpovídá dopravní situaci natož aby ji aproximovala. Další fází tak následuje oprava zmíněné křižovatky ve všech hlediscích, která již byla zmíněna. Tedy odstranění děr, trimování, úpravy logiky křižovatky a celková racionalizace.

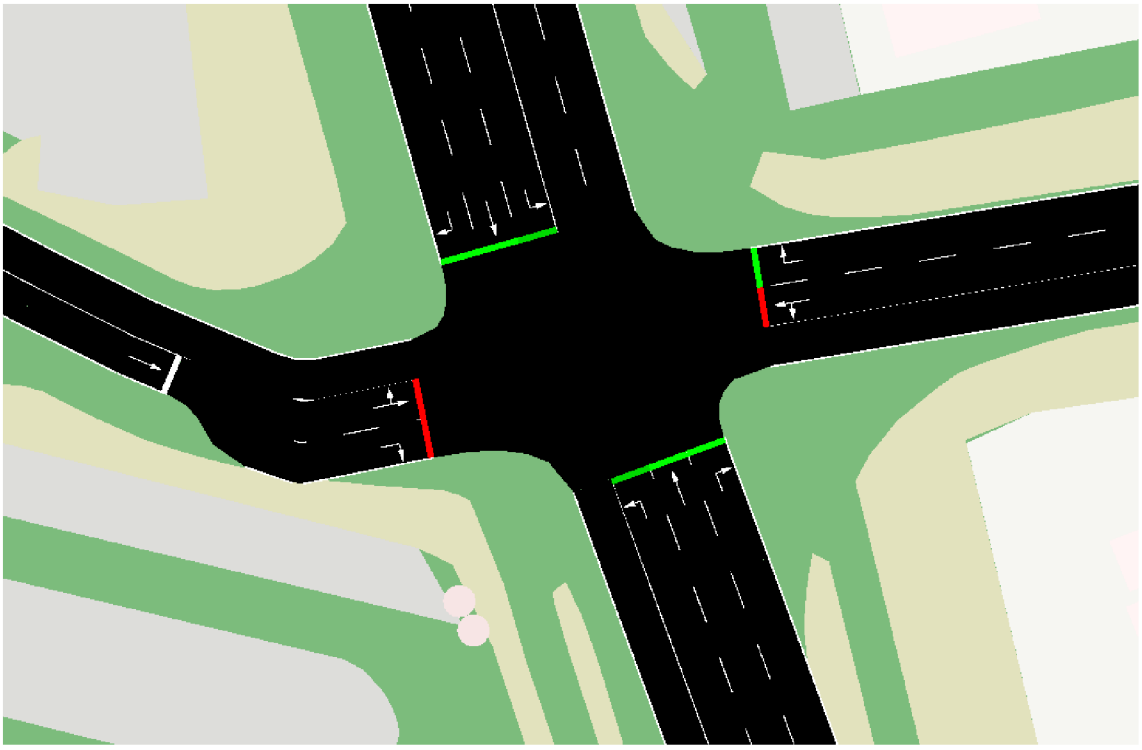
3.3.2 Manuální racionalizace modelu

Pro účely úprav importovaných dopravních sítí či jejich samotné vytvoření je používán již zmiňovaný nástroj NETEDIT. Software nabízí širokou škálu možných nastavení.

Obr. 14: Souhrn hlavních nástrojů pro úpravu dopravní sítě.

Zleva se jedná o nástroj Inspect, který podává informace o libovolných entitách uvnitř modelu. Každá spojnice má unikátní ID, pomocí kterého lze v přiložených souborech například nastavit proud aut na této spojnici začínající či končící. Navíc jsou přehledně zobrazeny vstupy do libovolné křižovatky, zde uzavřeny do parent – child struktury. Dalším nástrojem je Delete, velmi užitečný pro manuální redukci redundatních dopravních objektů. Následuje Select, který dává možnost společného označování a propojování jednotlivých elementů. Dále Move, jehož rozhraní je zobrazeno na obr. 13 pomocí malých růžových puntíků. Poskytuje možnost posouvání, natáčení a tvarování libovolné dopravní spojnice. Pokračuje Create, který dokáže vytvářet nové spojnice. Jeden z nejduležitějších nástrojů je zobrazen jako červené čáry na černém poli, Connection. Jednak se po přepnutí do tohoto módu zkompiluje model a vytvoří propojení ve všech křižovatkách. Jednak se také otvírá možnost pro editování těchto spojů. Významnými jsou ještě dva následující, Prohibition a Traffic light. První ze jmenovaných zakazuje určitým typům dopravy vstup, využívá se u tvorby pěších zón či cyklopruhů. Druhý nastavuje logiku světelných křižovatek, definici stavů a jejich jednotlivé délky.

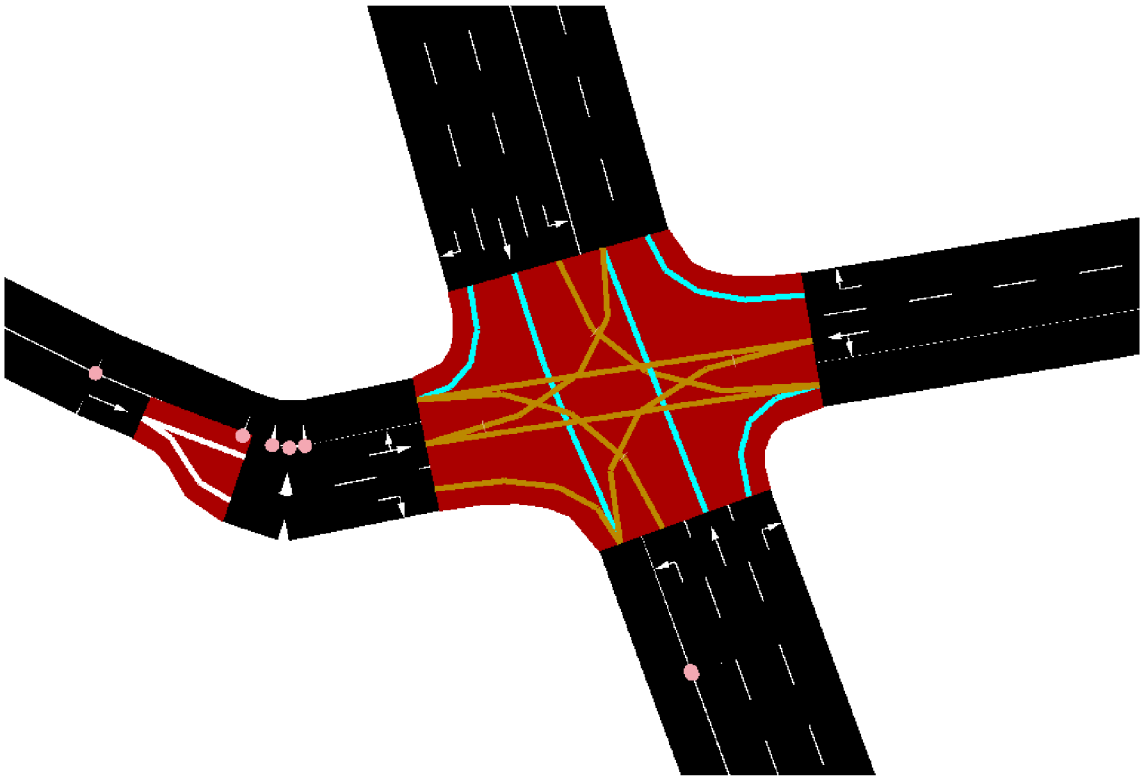
Primární obtíží byla nedostatečně dobrá inicializace křižovatky jako celku. Ony ostrůvky, které se zde ve skutečnosti nachází, se promítly do modelu buď špatně nebo vůbec. Dalším úskalím se ukázala vlastnost nástroje NETEDIT, a to nemožnost úpravy červené plochy. Uživatel sice může spojnice křižovatky libovolně natáčet i posouvat, jakmile však dojde ke kompilaci spojení jednotlivých cest, z červeného výseku nelze nic vylučovat ani jej jakkoli editovat. Aby dále fungovala logika světelné křižovatky, nebylo možné celou situaci modelovat jako více než právě jedno křížení. V neposlední řadě došlo k ořezání redundantních dopravních spojů a celkové racionalizaci. Ta spočívala primárně v manuálním nastavení všech možných propojení jednotlivých spojnic. V několika případech bylo také zapotřebí přidání jízdních pruhů, jelikož z důvodu nepřečtení dopravních ostrůvků si prostředí SUMO vyhodnotilo spojnici jako neobvykle širokou. Namísto toho však onen ostrůvek dělil odbočovací pruh od toho průběžného.



Obr. 15: Model křižovatky vytvořen za pomoci manuálních úprav.

Takto upravený model již plní svůj účel, přestože zde stále dochází k aproximaci. Hlavním úskalím je nemožnost vytvoření přesně uživatelsky definovaných trajektorií uvnitř křižovatky. Auta tak přejíždí oblasti které jsou v reálném rozložení zabrána právě neprůjezdnými úseky, tedy místy dopravních ostrůvků. Jedná se však spíše o záležitost kosmetickou, neboť na ráz silničního provozu nemá tato skutečnost vliv. Poslední významnou změnou, kterou si model prošel, bylo zahrnutí vzniku samostatného pruhu ve směru z východu na sever. Tato skutečnost sice bohužel nešla simulovat bez zahrnutí tohoto odbočovacího pruhu do logiky světelné křižovatky. Naštěstí však prostředí SUMO dává velkou svobodu v definici logiky libovolného funkčního komponentu modelu. V každé jednotlivé fázi, která se střídá v cyklu pro ovládání světelné křižovatky, tak bylo rozhodnuto o zelené pro směr z východu na sever. Funkčně tak dochází k velmi přesnému přiblížení se charakteru průjezdu křižovatkou. Přesto však v reálném provozu tento pruh ani světelným značením vybaven není.

Aby byla analýza proběhlých změn dokončena, ještě je rozebrána vnitřní logika uzlu, a to na obr. 16. Světle modré spojnice ukazují aktuální zelené fronty podle přetrvávající fáze v době vytvoření obrázku. Dále si lze povšimnou pouze jediné ústřední cihlové plochy, došlo tak ke spojení domnělých jednotlivých křižovatek. Dále byly upraveny délky hran křižovatky, aby lépe odpovídaly realitě.



Obr. 16: Zobrazení jednotlivých propojení a oblastí křižovatky v nástroji NETEDIT.

3.3.3 Dopravní zátěž

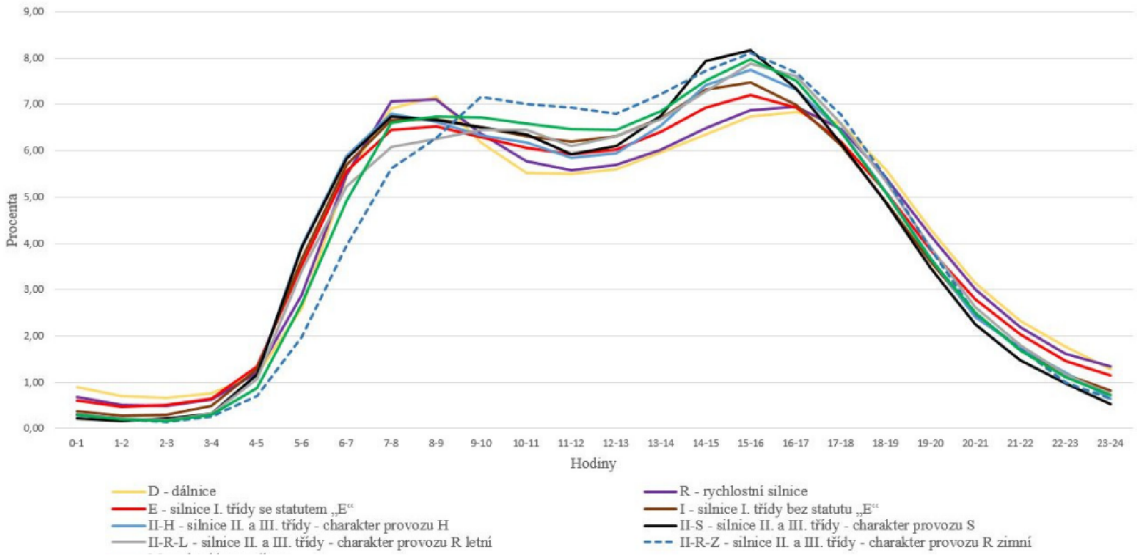
Ke kompletnímu dokončení modelu je nutné zahrnout do diskuse jeho účastníky, tedy samotnou dopravní zátěž. Jednou z neodmyslitelných vlastností dopravní sítě je situace, kde dochází k asymetrii dopravní zátěže. Stavu, kdy špičky poptávkových toků jednotlivých spojnic křižovatky jsou odděleny v čase. Příkladem je zhuštění dopravy v ranních a odpoledních hodinách, typicky poblíž začátku a konce standardní pracovní doby. Zároveň se však jedná o silně nedeterministickou úlohu z hlediska predikce hustoty dopravy jako celku, tedy předvídání dopravních toků v jednotlivých směrech. Vzhledem ke skutečnosti, že nejvyšší poptávka po efektivním řízení je právě ve špičkách. Další úvahy o realitě vypovídající aproximaci dopravy se vztahují k těmato dvěma špičkám.

Tato skutečnost byla do modelu zahrnuta za pomoci vytvoření dvou složek dopravy, pracovně nazvané jako statická a dynamická. První zmiňovaná má za úkol nastínit dopravní toky vyplývající z geografického umístění křižovatky. Směrem na sever se nachází podstatná část města, směrem na jih jedna jeho část. Západní cestou se nachází několik obchodů a jízdou na východ je možné nalézt jak připojení k E55, tak právě exit z ní. V ranních hodinách tak převládá doprava z východu, majorita směřující severně, minorita jižně. Odpoledne převládá doprava ze severu, přičemž majorita směřuje východně a minorita západně, tedy za účelem nákupu. Statický soubor obsahující

jednotlivé dopravní toky pracuje ve dvou stavech: ranní nebo odpolední špička. V každém z nich jsou nastaveny základní hodnoty dopravních toků zmíněnými směry.

Druhá složka dopravy, pracovně nazvaná jako dynamická, tvoří většinu simulovaných aut. Jedná se opět o XML soubor obsahující cesty, nyní však jednotlivých aut. Spojnice, na které se objevují, i na které svoji cestu dokončují, je vybrána pseudonáhodně. Důraz je kladen na vytváření aut v koncových spojnicích modelu, neobjevují se tedy například těsně před křižovatkou či za ní. Generování takové dopravy je umožněno s využitím souboru randomTrips.py, který je součástí instalace software SUMO. Jeden z mnoha nástrojů poskytnutých vývojáři, který při správné inicializaci a podrobném pročtení dokumentace je schopen velmi detailní randomizace na přání uživatele.

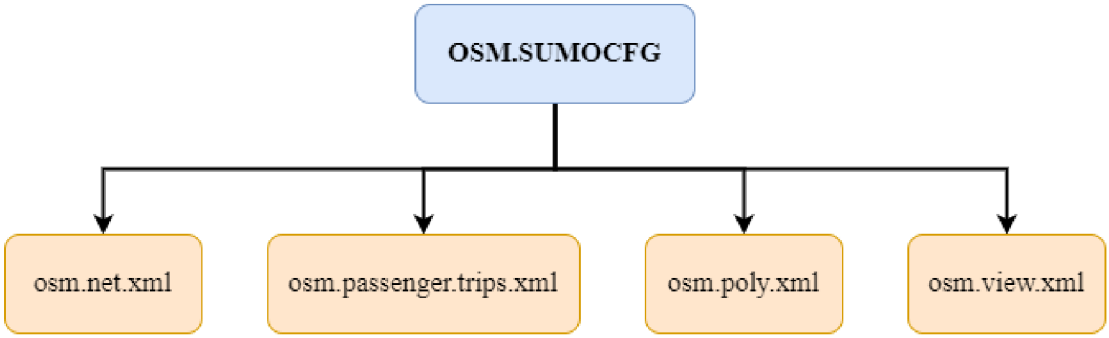
Rozdělení simulace do dvou schémat je podpořeno obrázkem 17. Jednoznačně vykresluje trend vzniku dvou hlavních dopravních špiček, a to jedné v ranních hodinách, druhé v těch odpoledních.



Obr. 17: Rozložení dopravy v jednotlivých hodinách pro různé druhy komunikací [27].

3.4 Datová struktura

Po instalaci a provedení nezbytných kroků, kterými jsou například vložení několika cest do systémových proměnných, je možné zapnout samotný simulační software, tedy SUMO. Po importování mapy či jakémkoli jinak zvoleném druhu inicializace prostředí lze zapnout samotný grafický software, sumo-gui. Po spuštění a inicializaci simulace se ve složce, kde se nacházely data modelu, objeví několik nových souborů, všechny ve formátu XML. Pro větší přehlednost byl vytvořen obr. 18.



Obr. 18: Skupina základních XML souborů, které simulace spoluvytváří.

Veškeré data obsahující soubory vytvářené spouštěcím souborem, tedy osm.sumocfg, jsou zobrazeny výše. Samotný spouštěcí soubor je rovněž psán v jazyku XML, jeho unikátní koncovka zaručuje automatické spouštění pomocí SUMO. Ve zkratce se dá účel jednotlivých souborů popsat jako:

- osm.net.xml – simulační síť
- osm.passanger.trips.xml – soupis projíždějících vozidel
- osm.poly.xml – tvary budov a důležitých míst
- osm.view.xml – nastavení zobrazení pro simulátor [28]

Následně jsou ještě založeny dva .bat soubory, run a build. Soubory s příponou .bat jsou dávkové soubory systému Windows. Jedná se o prosté textové soubory, které obsahují různé příkazy používané pro opakující se úkoly nebo pro spouštění skupin skriptů jeden po druhém. Run obsahuje několik parametrů pro spouštění simulace. Build je využíván u velkých zásahů do dopravní sítě, typicky úkony se spojováním křižovatek. Je tedy třeba druhý jmenovaný skript spustit se zavřeným simulačním prostředím, aby se provedené změny mohly korektně vepsat do již vytvořených XML souborů.

Veškeré nutné soubory pro správný chod programu tak byly založeny. Aby bylo možné jednak informace ze simulace získávat a jednak je také v reálném čase upravovat, je na uživateli, jaký postup zvolí. Každopádně je nutné využití vlastních skriptů, které se připojí k simulaci a efektivně ji řídí.

Z dostupných a rozšířených programovacích jazyků se jako nejschůdnější nabízí jazyk Python. Velmi dynamicky se rozvíjí a je zároveň uživatelsky přívětivý. Jeho největší devízou však zůstává velké množství komunitou vytvořených knihoven, které usnadňují komunikaci napříč různými uplatněními. Na základě výše zmíněných argumentů se tedy tento jazyk stal primárním rozhraním pro ovládání a čtení simulačních dat.

Základní úkoly, které musí vytvořený skript být schopen plnit, byly již nastíněny. Jedná se převážně o komunikaci s prostředím SUMO. Konkrétně tedy v iniciační fázi spuštění samotné simulace. Dále nastavení parametrů, jakými jsou množství dopravy, její rozvrstvení v čase i ploše. V každém kroku programu musí rovněž docházet k vyhodnocování účastníků provozu, tedy sbírat aktuální data simulace. Ty budou následně používány k vyhodnocování jednotlivých řídících strategií a jejich vzájemnému porovnání. V neposlední řadě je nutné simulace iterovat a zaznamenávat klíčové proměnné, aby bylo dosaženo statisticky významného množství dat.

Kostra takového skriptu již byla vytvořena, a obsahuje základní formu komunikace s prostředím SUMO [29]. Stěžejní část práce umožňuje knihovna TraCI. TraCI je zkratka pro "Traffic Control Interface". Zpřístupněním běžící simulace silničního provozu umožňuje získávat hodnoty simulovaných objektů a manipulovat s jejich chováním v reálném čase. TraCI používá k poskytování přístupu k sumo architekturu klient/server založenou na TCP. Samotná simulace je ukončována zadáním příkazu k uzavření. Simulace tímto není nutně zatížena maximální dobou trvání, nýbrž primárně přítomností účastníků silničního provozu [30].

```
***Starting server on port 51087 ***
Loading net-file from 'osm.net.xml' ... done (20ms).
Loading additional-files from 'tlslogic.xml' ... done (4ms).
Loading done.
Simulation version 1.12.0 started with time: 0.00
Simulation ended at time: 350.00
Reason: TraCI requested termination.
Performance:
 Duration: 8.13s
 TraCI-Duration: 0.35s
 Real time factor: 43.05
 UPS: 9083.333333
Vehicles:
 Inserted: 108
 Running: 0
 Waiting: 0
Statistics (avg):
 RouteLength: 373.48
 Speed: 6.03
 Duration: 84.44
 WaitingTime: 38.80
 TimeLoss: 56.11
 DepartDelay: 1.48

DijkstraRouter answered 108 queries and explored 7.81 edges on average.
DijkstraRouter spent 0.00s answering queries (0.00ms on average).
```

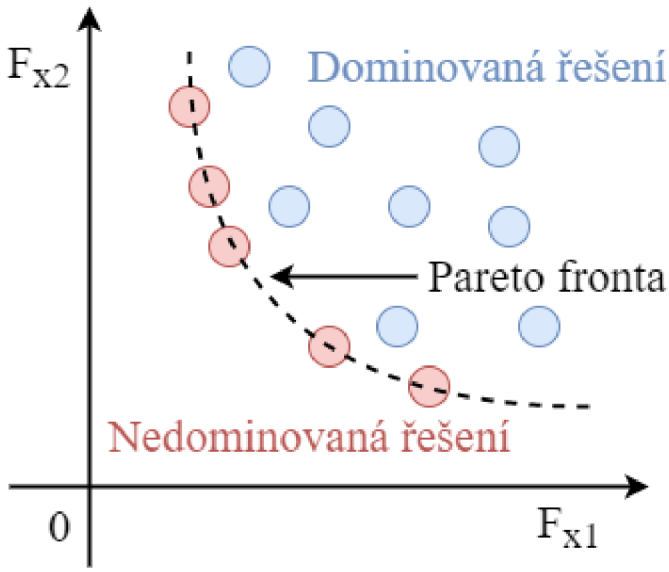
Obr. 19: Konzolový výstup po ukončení simulace s pomocí knihovny *TraCI*.

3.5 Definice optimalizačního problému

Dosud bylo předmětem práce primárně vytvoření modelu a jeho testování. Došlo k velmi slibné aproximaci skutečné křižovatky. Za pomoci přiložených skriptů a vlastní práce došlo k rozvrstvení dopravy. Ta má náhodnou a pevnou složku, dochází tak k fenoménu dopravního přetížení ve stejné podobě, v jaké k němu dochází v reálném provozu. Dále je však generována složka náhodná, která zaručuje hledání neprimitivního řešení, tedy nikoli takového, který vyhovuje pouze pro staticky dané dopravní toky. Aby byl zaručen determinismus úlohy, bylo nutné onu proměnlivou složku vhodně usměrnit. Příliš rozkmitaná soustava by buď neměla tendenci konvergovat k optimu či by jeho hledání nemuselo vůbec vést k řešení.

Parametry, na základě kterých jsou vedeny další úvahy o optimalizaci, jsou dva: průměrná délka čekání (avg waiting time) a maximální délka čekání (max waiting time). Jedná se o hodnoty v sekundách a představují časový úsek, po který dopravní účastníci stojí. Maximální délka čekání byla přidružena právě z toho důvodu, aby řídící algoritmy neměly tendenci sklouznout například k významnému odříznutí nějaké křižovatkové spojnice na úkor těch ostatních. Sice by tak v období dopravních špiček nejspíše došlo ke zvýšení dopravního toku, vedlo by však také k významné frustraci a celkové nespokojenosti odstřižených účastníků.

Vzhledem k tomu, že dochází k optimalizaci dvou parametrů současně, konkrétně minimalizaci, nazýváme definovaný problém jako multi-objektivní. Úlohy tohoto typu se obecně řeší hledáním k-tice nedominovaných řešení, ze kterých se nadále selektuje to optimální. K lepšímu pochopení byl vytvořen obrázek 20:



Obr. 20: Schéma řešení multi-objektivního optimalizačního problému – minimalizace.

Na obrázku výše je zachycen graf dvou proměnných, v našem případě například závislost maximální době čekání na průměrné době čekání. Naše optimalizační úloha je minimalizace. Hledání nedominovaných řešení spočívá v nalézání takových hodnot obou těchto kriteriálních funkcí, u nichž by zlepšení jedné z těchto hodnot nutně znamenalo zhoršení té druhé. Spojnice bodů, které splňují tuto vlastnost, se nazývá Pareto fronta. Občas se lze setkat také s pojmem Pareto optimální, což popisuje stejnou situaci. V grafické interpretaci minimalizačního problému tak hledáme body ve 2D soustavě, obecně se však jedná o hledání k-tic nedominovaných řešení.

Objekt zájmu je definován, tyto hodnoty jsou však pouze výstupy simulace. Aktuátory, tedy parametry, pomocí kterých lze dané proměnné minimalizovat, skýtá samotná světelná křižovatka. Její program, tedy posloupnost jednotlivých fází, se zpravidla používá jako ta, která byla představena v úvodní části práce. Volitelnými proměnnými tak zůstávají délky jednotlivých zelených front.

V období zvýšeného provozu z města pak bývá pro interesované směry vhodné zavést vyklizovací zelenou šipku. Zůstávají tak tři volitelné proměnné, a to délky tří zelených front. Jedná se o zelenou pro sever-jih, vyklizovací šipka sever-jih a východ-západ. Optimální nastavení těchto tří proměnných má za následek minimalizaci celkové čekací doby a průměrné čekací doby. Pro tyto parametry byly zavedeny omezující podmínky. Jednak příliš rychlé přepínání nesvědčí plynulosti provozu, jednak případní chodci musí být schopní v době své zelené nejenom přejít silnici, ale navíc ji ještě uvolnit pro odbočující auta. Z tohoto důvodu byla zvolena minimální délka trvání patnáct sekund pro průběžné zelené, tedy první a třetí parametr. Dále je minimální doba trvání zvolena tři sekundy pro vyklizovací šipku, druhý nastavovaný parametr, neboť má charakter uvolnění křižovatky k hladkému přechodu do další fáze světelné křižovatky.

Následují charakteristiky jednotlivých algoritmů a jejich implementací. Simulace jednotlivých nastavení jsou časově náročné. Jedna iterace, tedy testování konkrétního nastavení křižovatky s využitím knihovny *TraCI*, se délkou pohybuje v řádu jednotek sekund. První zvolený algoritmus, již zmiňovaný Grid search, musí těchto výpočtů provést tisíc, aby bylo získáno dostatečné množství dat. Podobně tomu je u genetického algoritmu.

3.6 Prohledávání v mřížce

Prvním algoritmem pro získání optimálního nastavení světelného značení semaforu je již zmiňovaný Grid search neboli prohledávání v mřížce. Všechny tři volitelné parametry se tak rozdělí do intervalů, v nichž pro každou možnost dojde k vyhodnocení nejvyšší a celkové čekací doby. Následně se vybírají nedominovaná řešení.

Nejsložitějším úkolem se ukázala samotná implementace, a to způsob iteračního přepisu délek fází světelných signálů. Při běhu simulace není možné měnit délky zelených front dynamicky, pouze měnit již předdefinované programy či prodlužovat zbývající dobu trvání aktuálního stavu semaforu. Dokumentace, která je volně dostupná, sice

popisuje přípustné metody knihovny *TraCI*, u většiny však není vysvětlen formát vstupu či naprosto chybí popis metody. V jistých případech bylo dokonce jednodušší prohledávání souborů patřících nainstalované knihovně a hledání jejich metod v jednotlivých skriptech, jelikož zřídka obsahovala i definice svých vstupů.

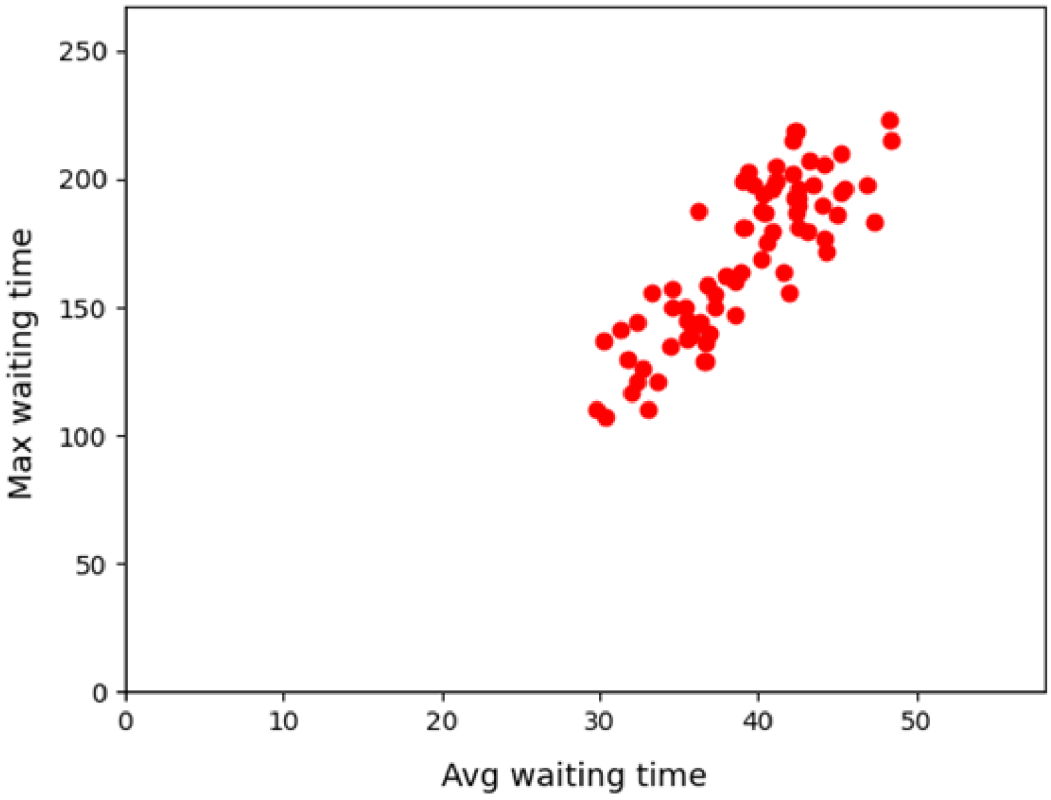
První způsob implementace přepisu dat byl realizován s pomocí datové struktura logic. Jedná se o strukturu, která se nejvíce podobá listu. Samotná dokumentace pak navrhuje, že nejlepším postupem pro definici vlastní logiky je první si o tento logic požádat metodou *traci.trafficlight_getAllProgramLogics().* Tím je získán přehled o správné syntaxi a celkovém uspořádání jednotlivých parametrů. Toto volání má za následek iterační generování všech logik s pěti parametry. V tomto případě jediné logiky. Ve chvíli kdy však dochází k pokusu definovat novou logiku pomocí metody *traci.trafficlight_setProgramLogic(),* povoleny jsou pouze tři parametry. Problémem je nejspíš vývoj samotné knihovny, kdy různé metody a funkce postupně nejsou kompatibilní se zbytkem knihovny. Podobně tak i tyto dvě funkce, přestože se podle dokumentace má jedna používat pro generování té druhé, již nepodporují stejný formát, tudíž se dají jen stěží využít. Podobné způsoby, tedy využívání dalších nativních metod knihovny byly neúspěšné, jelikož buď využívaly strukturu logic, nebo mohly například měnit pouze aktuální dobu trvání zelené fronty, nikoli přepisovat tyto hodnoty do přiložených XML souborů.

Další možností, která se nabízela, bylo přepisování samotného XML souboru, který obsahuje logiku křižovatek, a to vždy po uvolnění souborů při ukončení chodu *TraCI* smyčky. Nejprve tak byl vyčleněn přídavný soubor *tlslogic.xml,* který neobsahuje další tisíce řádků popisují funkční vazby modelu. Sumo rovněž preferuje přiložené soubory nad těmi přikládanými povinně, tudíž je logika semaforu automaticky brána odsud. Dalším úskalím se ukázalo netypické uspořádání dat uvnitř XML souborů. Jednotlivé subelementy totiž nejsou jako obvykle děleny na tagy a atributy, tedy rozložení, kdy na jeden tag připadá jeden atribut. Tagem je v tomto případě phase, neboli fáze, atributy jsou však hned dva, duration a state. Bylo tedy opět třeba pátrat po metodě nějaké ze známých knihoven zabývajících se přepisováním XML souborů. Standartní tutoriály a materiály totiž pracovaly pouze s rozložením jedné hodnoty na řádek. Dokud tedy nebylo definitivní zavržení postupu číslo jedna, bylo očekáváno, že je podobná editace náročná, což se ukázalo jako nepravdivé.

Cestou se ukázalo být použití knihovny *xml.etree.ElementTree,* která obsahuje spoustu nástrojů na manipulaci s xml soubory v prostředí python. Knihovna je velmi uživatelsky přívětivá a pomocí několika málo příkazů je schopna vytáhnout veškeré informace z libovolného xml souboru. Nyní již stačilo správně zvolit syntaxi. Nejlépe se osvědčila metoda nejprve promazání stávajících dat a následné nastavení podle nachystaných délek jednotlivých zelených front. Parser užívaný v sumo knihovně si poradil s reprezentací na jeden řádek, přestože se jedná o méně líbivou grafickou úpravu.

Řídící skript je tak schopen měnit vstupní XML soubory, pomocí čehož lze docílit optimálního nastavování křižovatky v offline módu, tedy mimo běh samotné simulace. Jedná se však o výpočtově náročnou záležitost, neboť každý problém narůstá exponenciálně s každou přidanou řídící proměnnou. Ani pro velmi pečlivé analýze přiložené dokumentace nedošlo k nalezení informací vedoucí k detekci metody k získání statistických dat chodu programu. Metod poskytuje knihovna TraCI opravdu spoustu, přesto žádná nevede k vyčtení důležitých simulačních dat, jakými jsou průměrná čekací doba (avg waiting time) a celkový počet aut. Standartní inkrementační metody jsou nepoužitelné, jelikož při běhu simulace není pevně dán počet iterací. Smyčka je ukončena až po opuštění simulace všemi auty. Velmi elegantním se tak stalo řešení reportování uvedených statistik do zvláštního souboru *statoutput.xml*. Odtud pak podobným způsobem, jako u nastavování zelených front, je parsován obsah a vyčteny hodnoty celkového počtu aut a průměrné doby čekání.

Výše popsaným způsobem je tedy spouštěna simulace s částečně randomizovanou dopravou, na které je prováděno vyhledávání optimálního nastavení světelné křižovatky. Vzhledem k výpočetní náročnosti byl stavový prostor omezen na sedm set možností. Dvojice řešení průměrné čekací doby a celkové čekací doby jsou následně vykresleny za pomoci knihovny *Matplotlib*.

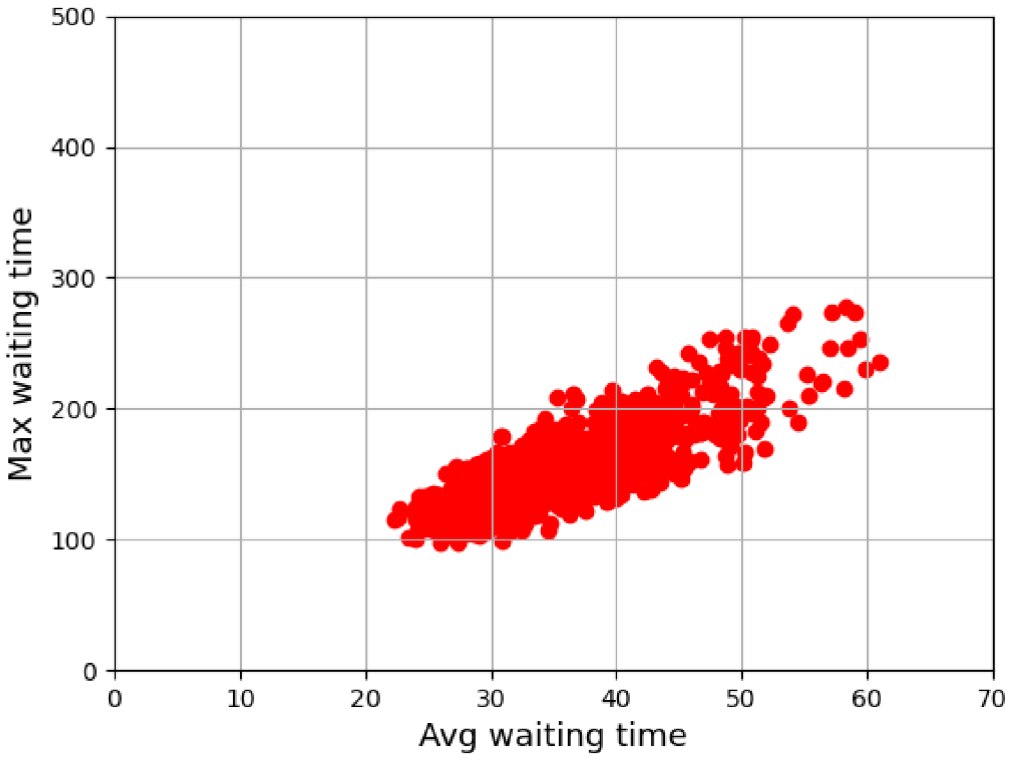


Obr 21: Výstup algoritmu prohledávání v mřížce.

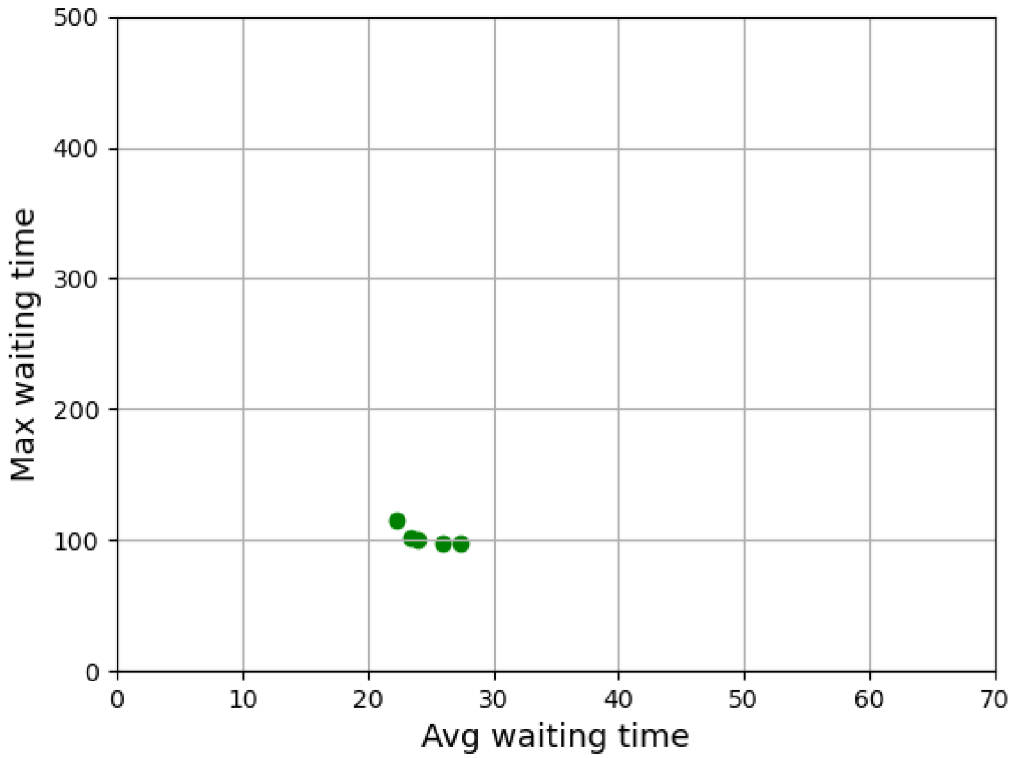
Na obrázku 21 je zobrazen důraz na detail ve smyslu jednotlivých bodů, v dalších srovnáních je však pro lepší čitelnost přidána mřížka. Finálním krokem algoritmu je detekce Pareto optimálních řešení. Z toho důvodu je v implementaci zařazena speciální třídící funkce schopná vyfiltrovat všechny ostatní body. Zpětně je pak pomocí znalosti logiky kódu vyhledáno nastavení světelné křižovatky, které vedlo k získání optimálního řešení.

3.6.1 Ranní doprava

Pro hledání optimálního nastavení délek zelených front světelné křižovatky bylo tedy využito prohledávání v mřížce, algoritmu Grid search. Minimální hodnoty byly vybrány s ohledem na aproximaci chodců, stejně tak jako s ohledem na plynulost a hladkost provozu. Velikost mřížky byla volena na tři sekundy a došlo k variaci deseti hodnot pro všechny tři nastavované parametry. Celkově tedy následuje vykreslení tisíce řešení. Poté je zobrazen detailnější graf zachycující pouze ta Pareto optimální.



Obr 22: Nalezená řešení pro ranní dopravu algoritmem Grid search.

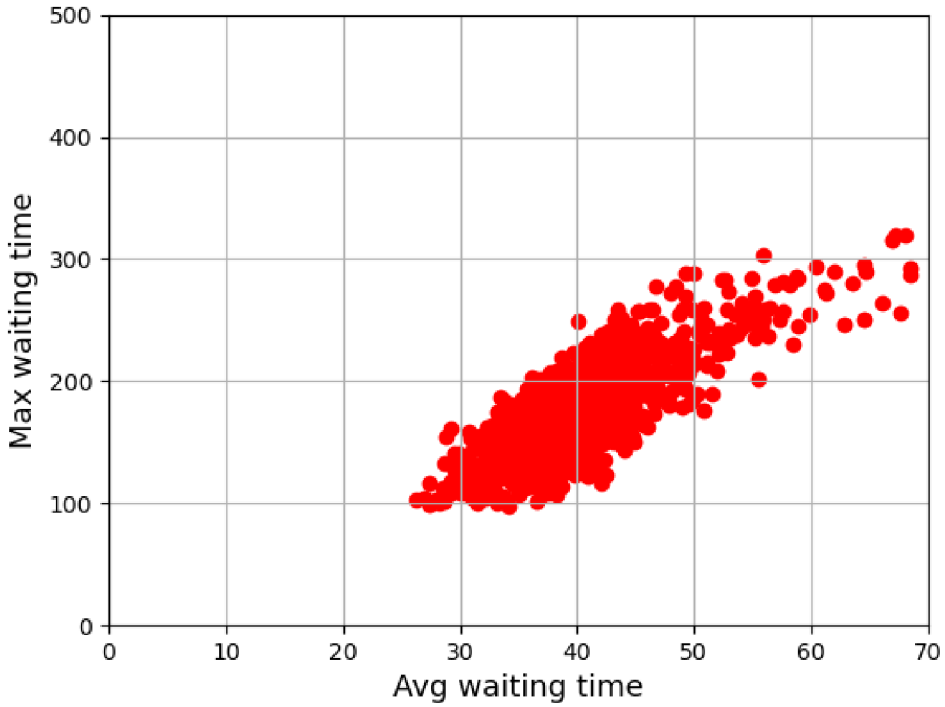


Obr 23: Vybraná Pareto optimální řešení z výše uvedeného vzorku.

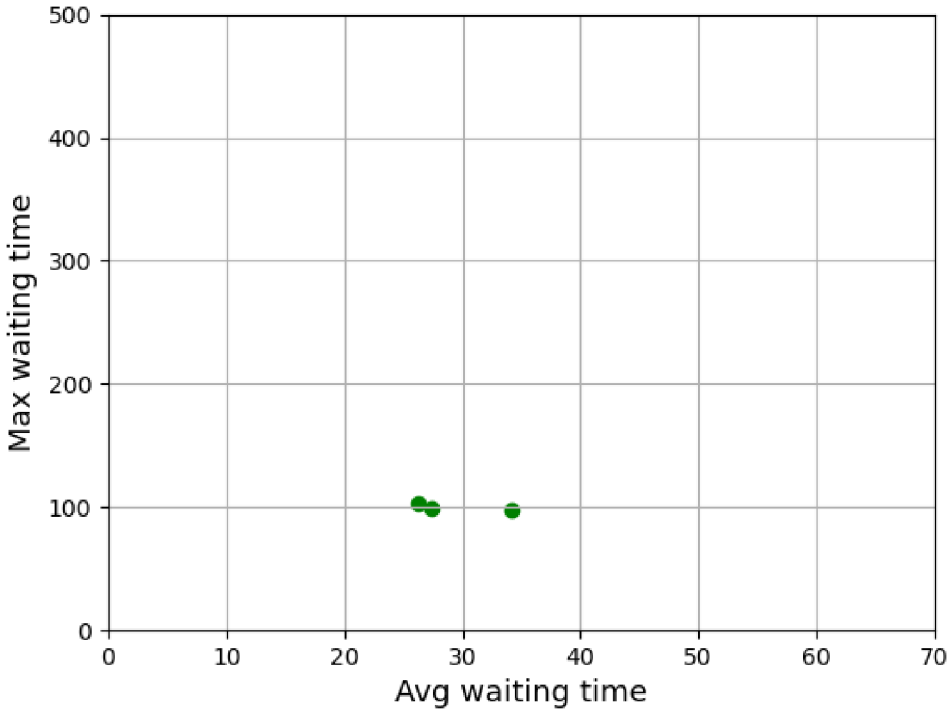
Nalezené Pareto optimální hodnoty jsou rovněž přepočítávány zpět na interpretaci pomocí vstupních parametrů. Trojice vstupních parametrů je tak ukládána samostatně do textového souboru. Optimálním nastavením byly shledány následující posloupnosti vstupů: [15, 4, 21], [17, 4, 17], [19, 6, 19], [15, 4, 17], [19, 5, 23].

3.6.2 Odpolední doprava

Pro získání optimálního nastavení světelné křižovatky ve schématu odpolední dopravy byla pozměněna statická část dopravy, která se nachází v souboru *osm.passenger.trips.xml*. Co do objemu aut se jedná o totožné scénáře, mění se však starty i cíle těchto toků aut. Následný scénář byl opět otestován s volbou mřížky o velikosti tři sekundy. Opět se iteruje přes tři parametry, každý s mocností deseti možností. Dochází tak k vykreslení tisíce jednotlivých stavů. Nejprve dojde k zobrazení prohledávání všech, následuje výběr Pareto optimálních řešení.



Obr 24: Nalezená řešení pro odpolední dopravu algoritmem Grid search.



Obr 25: Vybraná Pareto optimální řešení z výše uvedeného vzorku.

Nalezené Pareto optimální hodnoty jsou opět přepočítávány zpět na interpretaci pomocí vstupních parametrů. Trojice vstupních parametrů je tak ukládána samostatně do textového souboru. Optimálním nastavením byly shledány následující posloupnosti vstupů: [15, 10, 21], [27, 8, 27], [36, 9, 42]

3.7 Genetický algoritmus

Druhým algoritmem pro získání optimálního nastavení světelného značení byl zvolen genetický algoritmus. Tento přístup byl zvolen z důvodu obvykle silné konvergence k optimálnímu řešení, které je ovšem vykoupeno vyššími výpočetními nároky. Rychlost i přesnost optimalizace pomocí genetického algoritmu je závislá na parametrech jakými jsou počet generací a počet potomků. Dohromady taky spoluurčují celkovou výpočetní náročnost.

Genetickým algoritmem, zvoleným pro aplikaci řízení světelné křižovatky, se stal algoritmus NSGA-II. Jedná se o druhou generaci takzvaného Non-dominated Sorting Genetic Algorithm, tedy Nedominovaného Třídícího Genetického Algoritmu. Využívá se v případech multiobjektivní optimalizace. Jak již bylo nastíněno, opět se optimalizují dva parametry, a to průměrná doba čekání a maximální doba čekání [31].

Základní implementace takového algoritmu byla zvolena podle [32]. Jedná se o skupiny funkcí, které mají být schopny efektivního prohledávání stavového prostoru za účelem rychlé konvergence. Společně s vyhodnocovací logikou jsou schopny optimalizace pro jednorozměrné úlohy. Těmito úlohami rozumíme problémy, ve kterých je aktuátor ve formě jediné proměnné. Logika modelu křižovatky však vyžaduje nastavování tří proměnných.

Vlastní implementace tedy spočívá primárně ve třech krocích. Zaprvé využití předchystaných funkcí a logiky v takové formě, která umožňuje vyhodnocovat trojrozměrné úlohy. Zadruhé propojení vyhodnocovaných dat se simulací. Jednak musí v každém kroku docházet k přepisování nastavených hodnot simulace, jednak musí docházet ke čtení simulačních dat k vyhodnocování správných kroků. Zatřetí vytvořit nástroje schopné detailního vykreslování. Interpretace je opět zvolena v prostředí Python.

První úskalí se ukázalo být majoritním oproti ostatním, neboť libovolný rozhodovací krok uvnitř definované metody či vnitřní logiky byl implementován pro čísla či pouze listy. Nyní byly ovšem jak samotná vstupní data, tak deriváty z nich, interpretovány buď jako listy, nebo listy listů, případně pomocí knihovny *Numpy* rovnou jako pole. Došlo tak v podstatě k přepracování celého souboru, kdy většina nástrojů byla využita spíše jako vzor, než aby byly využity nástroje samotné. Logika systému rovněž odpovídala optimalizaci pomocí jediné proměnné, což v mnoha rozhodovacích uzlech znemožňovalo jejich původní fungování.

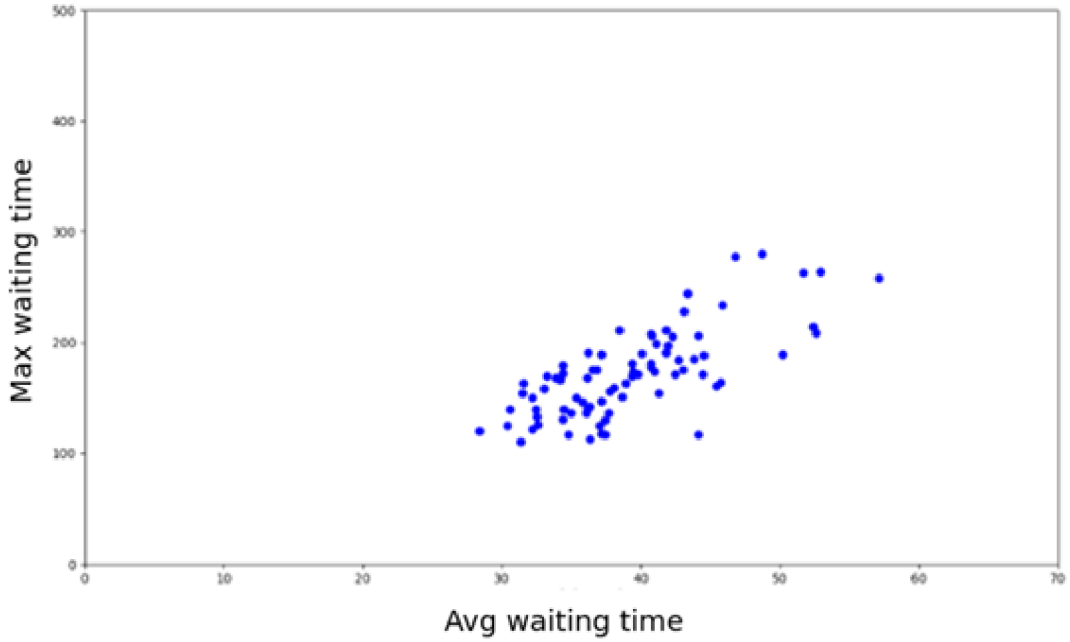
Druhým úskalím byla komunikace mezi simulací a tímto zmiňovaným řídícím skriptem. Problémy, které má knihovna *TraCI* již byly nastíněny výše. Naštěstí zde však došlo k využití poznatků s implementací vyhledávání v mřížce. Opět se tak vyčítají

i zapisují hodnoty parametrů vstupních i výstupních z vytvářených XML souborů. Zajímavostí se však stalo chování knihovny *TraCI* ve chvíli, kdy existovaly dva skripty, které ji byly schopny využít, v jednom adresáři. Docházelo totiž ke spouštění skriptu pro vyhledávání v mřížce uvnitř skriptu pro genetický algoritmus. Dokonce i bez toho, aniž by bylo cokoliv ze zmiňovaného skriptu importováno. Součástí nástrojů knihovny tedy musí být nějaká forma paměti, a to nejspíš odkaz na již proběhlá volání. Došlo-li tedy ke spouštění skriptu obsahujícího genetický algoritmus poté, co byl testován Grid search, opět se spustilo toto mřížkové vyhledávání. Celkové řešení se tedy nachází ve dvou oddělených adresářích z důvodu zamezení možné nekompatibility spouštěním různých skriptů za sebou.

Toto řešení se sice ukázalo jako jediné možné, skrývá v sobě však nepříjemnosti. Definice inicializačních parametrů, jakými jsou název křižovatky, jednotlivé zvolené stavy těchto světelných dopravních uzlů či parametry mající roli v inicializaci začátku samotné simulace nejsou sdílené mezi soubory. Pokud by tak uživatel chtěl měnit spouštěcí parametry nebo charakter dopravy v jednom z vytvořených XML souborů, je nucen tyto editace provádět dvakrát. Možným způsobem pro vyřešení tohoto problému by mohlo být vytvoření *TraCI* server struktury, která by striktně umožňovala procházení samostatného skriptu a poskytovala právo na vykonávání jako unikátní posuvný tiket.

K vykonávání přepisu i čtení hodnot tedy bylo opět využito XML souborů. K tomuto účely bylo definováno několik metod, které zpřehledňují tělo programu absencí redundantního opakování kódu. Přestože je knihovna *TraCI* silně optimalizována, doba nutná k jednomu takovému výpočtu se i s inicializací potřebných parametrů pohybuje v rámci jednotek sekund. Tato hodnota je závislá na charakteru simulované dopravy a tedy na dynamické složce vytvářených dopravních cest. Tyto cesty jsou vytvářeny skriptem *randomTrips.py,* a to způsobem, který již byl popsán.

Posledním problémem bylo samotné vykreslování, kde po vzoru minulého skriptu byla využita knihovna *Matplotlib.* Jednak je ukázán průběh hledání řešení napříč kroky. Navíc je rovněž vykresleno samotné řešení, ke kterým dospěli jedinci z poslední generace.

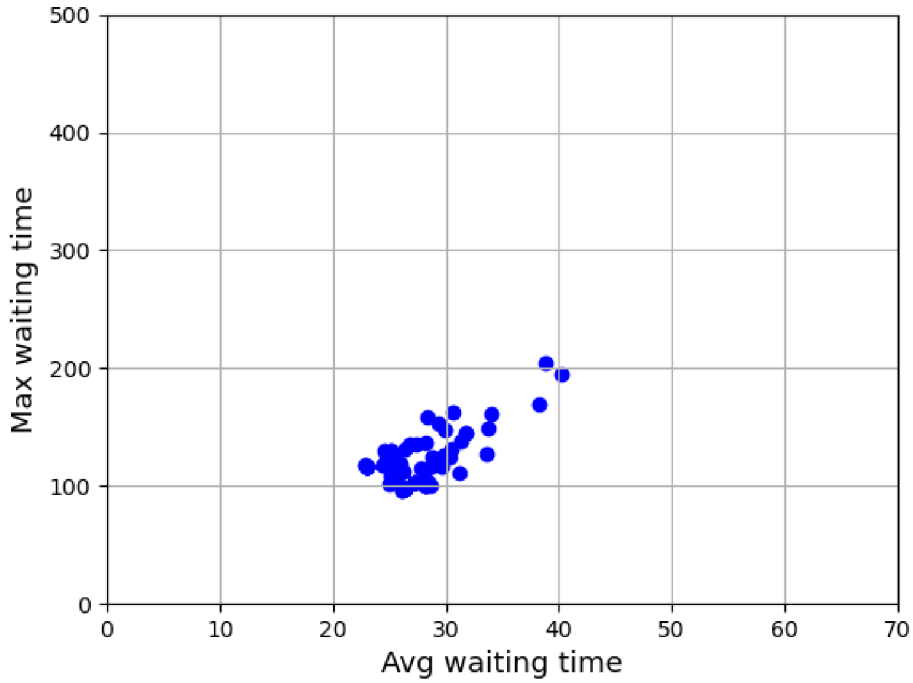


Obr 26: Ukázka průběhu řešení genetického algoritmu NSGA-II.

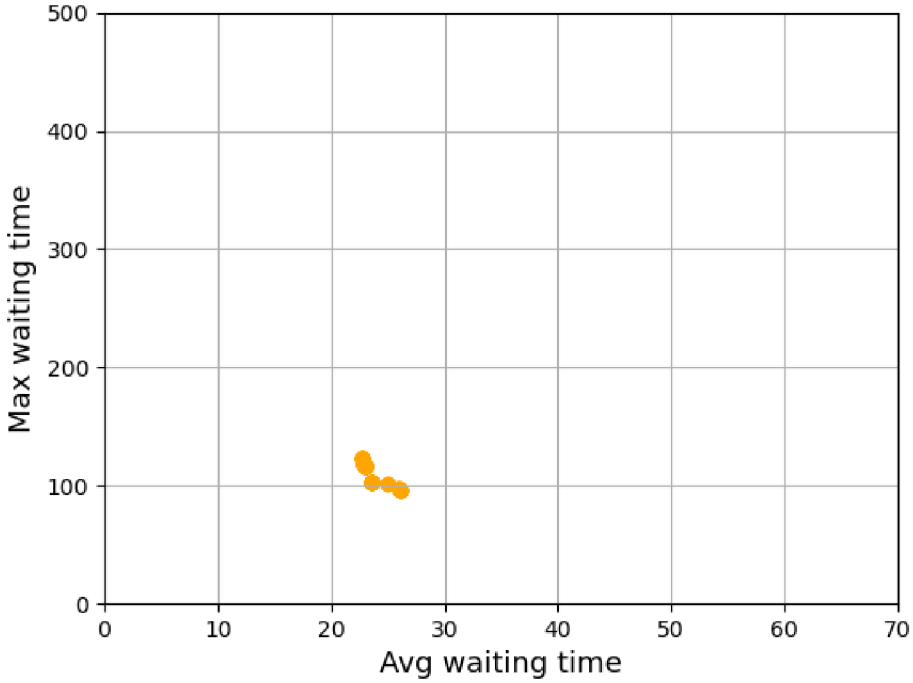
Na obrázku výše je kladen důraz na detail ve smyslu jednotlivých bodů, v dalších srovnáních je však pro lepší čitelnost zmenšeno rozlišení a přidána mřížka. Podobně jako tomu bylo v předešlém případě, i tato data je třeba vyfiltrovat tak, aby došlo k získání nedominovaných řešení. Opět je cílem získání trojice parametrů, které popisují optimální nastavení světelné křižovatky.

3.7.1 Ranní doprava

Prohledávání stavového prostoru bylo realizováno genetickým algoritmem NSGA-II. Vykreslování bodů zde probíhá v jednotlivých generacích, přičemž se algoritmus snaží vyhledávat lepší a lepší řešení. Pro simulační účely byla zvolena populace o velikosti sto a počet generací roven dvaceti. Průběh řešení zde nepůjde jednoduše interpretovat pomocí jediného grafu, jelikož je řešení získáváno iterativní metodou. Proto byla zvolena ukázka desáté generace, která je opět podpořena Pareto výběrem nejlepších řešení generace poslední.



Obr 27: Vykreslení desáté generace ranní dopravy genetického algoritmu.

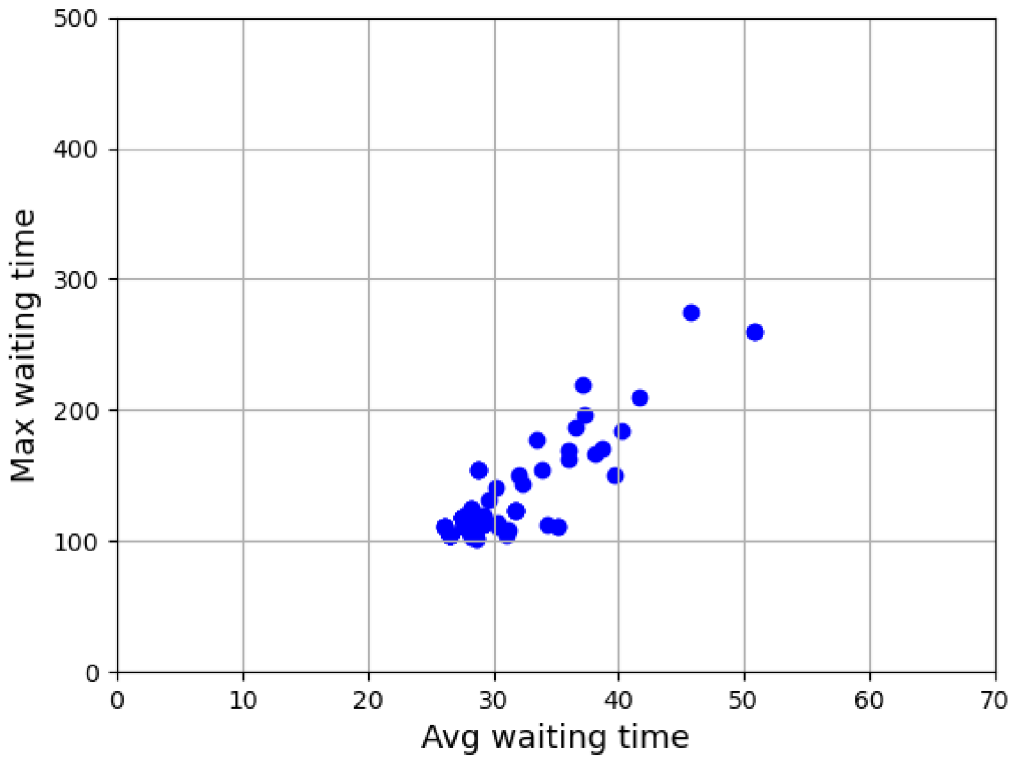


Obr 28: Výběr Pareto optimálních řešení z finální dvacáté generace.

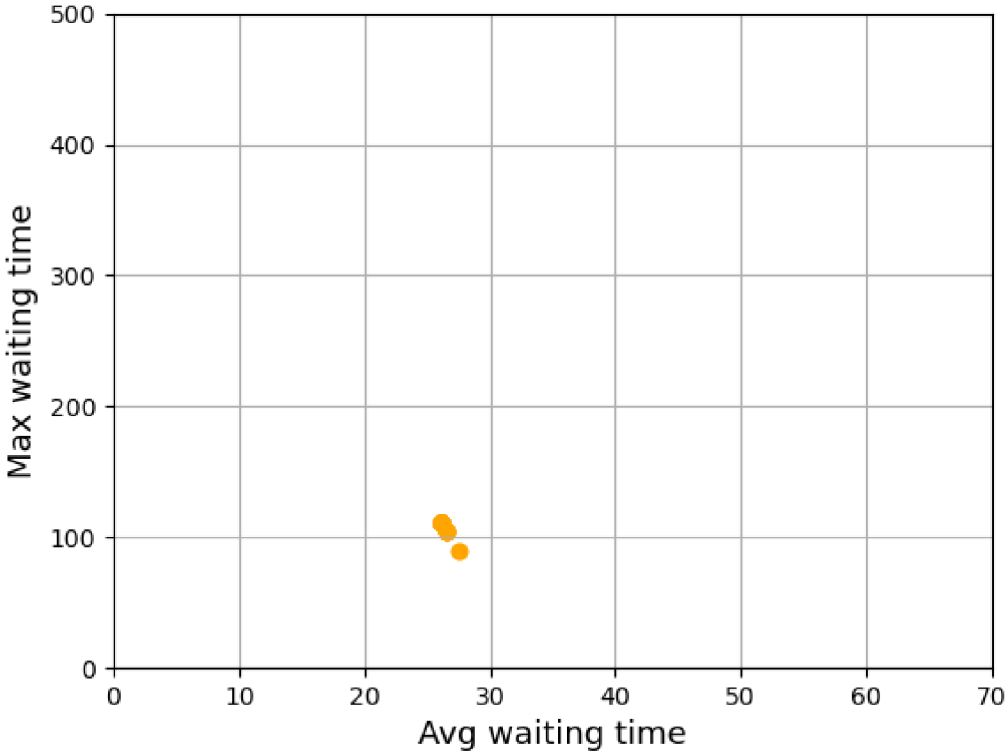
Nalezené Pareto optimální hodnoty jsou opět přepočítávány zpět na interpretaci pomocí vstupních parametrů. Trojice vstupních parametrů je tak, podobně jako u algoritmu prohledávání v mřížce, ukládána samostatně do textového souboru. Optimálním nastavením byly shledány následující posloupnosti vstupů: [15, 3, 16], [30, 5, 20], [15, 5, 16.], [24, 5, 16].

3.7.2 Odpolední doprava

Obdobně jako v předchozím případě dochází k přepnutí schématu dopravy, konkrétně k úpravě v souboru *osm.passenger.trips.xml*. K vykreslení a následnému ohodnocení je opět využíváno již osvědčené stovky jedinců v populaci a dvaceti generací. Kvůli iterativnímu charakteru výpočtu je zvolena desátá generace pro vykreslení a Pareto optimum z generace poslední.



Obr 29: Vykreslení desáté generace odpolední dopravy genetického algoritmu.



Obr 30: Výběr Pareto optimálních řešení z finální dvacáté generace.

Nalezené Pareto optimální hodnoty jsou přepočítány do trojrozměrné interpretace vstupních parametrů. Tyto hodnoty nastavení délek trvání světelných front u zkoumané křižovatky jsou ukládány do textového souboru. Optimálním nastavením byly shledány následující posloupnosti vstupů: [15, 12, 22], [16, 12, 18], [16, 11, 18].

4 ZHODNOCENÍ A DISKUSE

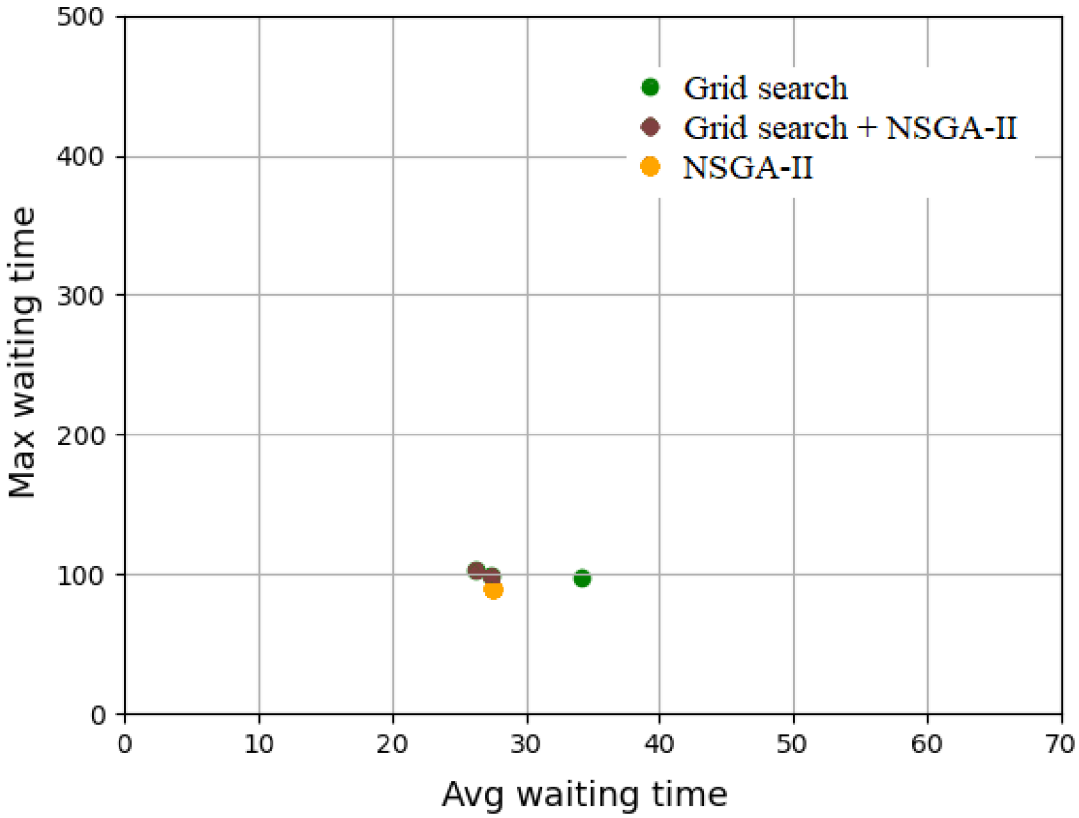
K získání optimálního nastavení světelných front zelené křižovatky byly tedy využity dva algoritmy, jednak prohledávání v mřížce, jednak genetický algoritmus. Nastavovány byly velikosti tří parametrů, reprezentující délky zelených fází uvnitř cyklu světelné křižovatky. Vyčítány byly dvě proměnné, a to průměrná doba čekání spolu s maximální dobou čekání. Jedná se tedy o trojrozměrný problém na vstupu a dvojrozměrný na výstupu.

Aby vznikla vypovídající reprezentace simulačních dat, práce využívá primárně srovnání v grafech dvou proměnných. Osy těchto grafů představují průměrné čekací doby a maximální čekací doby. Z těchto grafů je patrné rozdělení řešení uvnitř stavového prostoru. Jedná se o minimalizační úlohu, řešení nejblíže počátku jsou tedy obecně nejvhodnější. Jelikož se však jedná o optimalizační problém se dvěma účelovými funkcemi, vybírány jsou nedominovaná řešení, z nichž je následně zvoleno to optimální. Tato řešení společně tvoří takzvanou Pareto frontu.

Srovnávány jsou také nastavení odpovídající rannímu a odpolednímu charakteru dopravy. Pro tyto účely dochází ke změně takzvané statické složky dopravy, tedy nerandomizované části vyplývající z charakteru poptávky dopravní obsluhy v daném časovém okně. Aby byly výsledky porovnatelné mezi těmito schématy, nedochází ke změně dynamické složky dopravy ani ke změně logiky světelné křižovatky.

Vzhledem k již vysokému počtu proměnných parametrů, ve smyslu dynamické složky dopravy a množství délky ovládaných zelených front, je modelování chodců spíše nepřímou záležitostí. Jejich přítomnost se projevuje v definování minimálních délek průběžných zelených front. Díky tomu je zajištěna volnost přecházení, přičemž důsledek takové akce by výsledek ovlivňoval pouze v případě odbočujících. Jelikož již nyní dochází k přednosti pro auta projíždějící rovně, zdržení by primárně zasáhlo auta odbočující vpravo. Jedná o dopravní uzel na periferii dvou městských částí, zdržení by proto nebylo dlouhé. Přesto je nutné zde poznamenat onu aproximaci, ke které dochází.

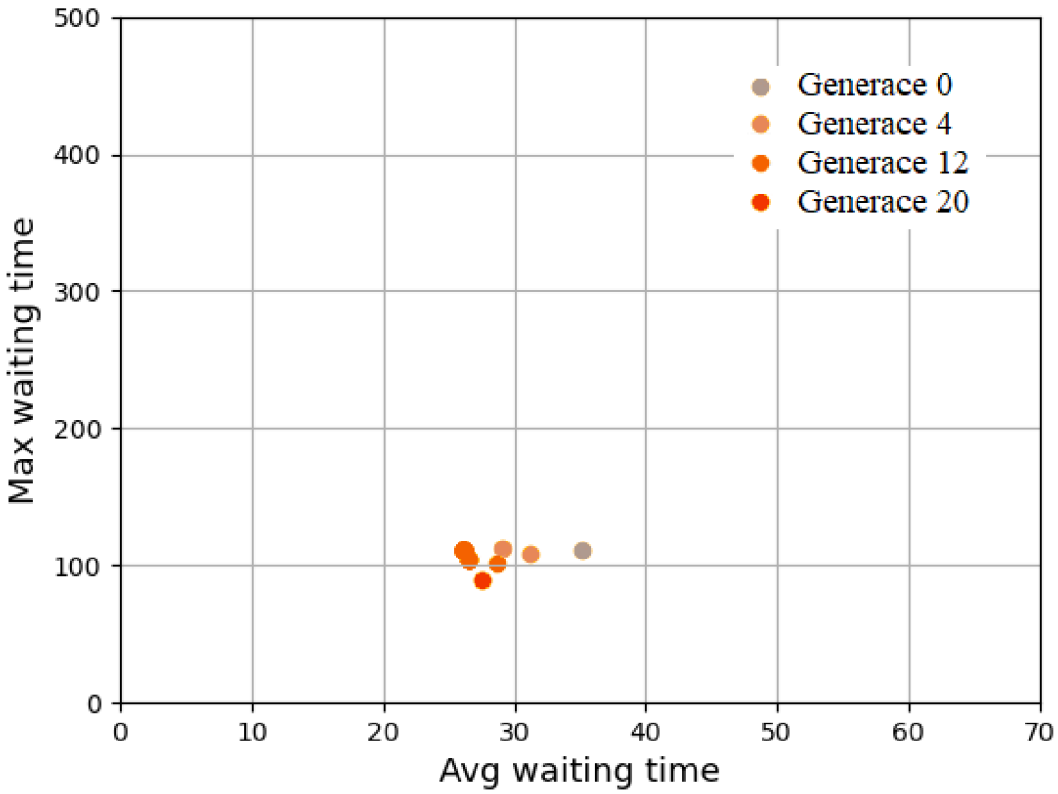
4.1 Srovnání pareto NSGA-II x Grid search



Obr 31: Vykreslení Pareto stavů algoritmů NSGA-II a Grid search v odpolední dopravě.

Grafická reprezentace nejlepších výsledků obou algoritmů v případě odpoledního schématu dopravy. Výsledek srovnání je velmi těsný, neboť oba algoritmy našly tři nedominovaná řešení, z nichž dvě se svými výsledky prakticky shodují. Třetí řešení je však výrazně lepší u algoritmu NSGA-II oproti algoritmu Grid search. Jedná se o očekávaný výstup, neboť aplikovaný genetický algoritmus disponuje silnějším aparátem na prohledávání stavového prostoru. Vzniká tak důvod dále analyzovat průběh algoritmu NSGA-II a to primárně s ohledem na množství iterací.

4.2 Srovnání Pareto NSGA-II v průběhu iterací



Obr 32: Zobrazení průběhu hledání Pareto optimálních stavů v rámci generací.

Z výše uvedeného grafu je patrná tendence zlepšování algoritmu NSGA-II v rámci jednotlivých generací. Každý odstín představuje ty stavy, které byly nově nalezeny. Pokud byl nějaký stav ponechán v Pareto frontě i v dalších generacích, není překreslen novým odstínem. Díky tomu je vidět přírůstek v čase, třebaže se do finální Pareto fronty zařazuje stav generace dvacet spolu se dvěma nedominovanými stavy z generace čtrnáct. Po dvacáté generaci nebylo až do padesáté generace objeveno žádné nové řešení. Ukazuje se tedy, že aproximační chyba, která model spolu se simulací doprovází, převyšuje míru zlepšování u generací dvacet a více. Vzhledem ke kompromisu, který je nutné dělat mezi výpočetní náročností a mírou přiblížení se ke globálnímu optimu, tak není přínosné testování generací v řádu vyšších desítek.

4.3 Srovnání ranní x odpolední optimum

Analýzou výsledků simulování ranního a odpoledního lze získat značný rozdíl hodnot optimálního nastavení světelné křižovatky. Pro srovnání jsou využity data z algoritmu NSGA-II, jelikož poskytl lehce lepší výsledky vyhledávání. Nalezené parametry nastavení jsou tedy následující:

- [15, 3, 16], [30, 5, 20], [15, 5, 16.], [24, 5, 16] pro ranní dopravu

- [15, 12, 22], [16, 12, 18], [16, 11, 18] pro odpolední dopravu

Z těchto dat si lze povšimnout několika současných trendů. V rámci ranní dopravy je poptávka po zelené odbočovací šipce, reprezentované prostřední hodnotou, velmi nízká. Jedná se totiž o řízení odbočovacího pruhu ve směru ze severu na východ, tedy ve směru z centra města do na obchvat, zmiňovanou silnici E55. Dále dochází k dominanci významu první zelené fronty, tedy směru sever-jih. Hlavním důvodem je fakt, že přestože majorita ranní dopravní poptávky pramení z východu na sever, tento proud je vybaven samostatně vytvářeným dopravním pruhem. Dochází tak k rovnoměrné disipaci této poptávky bez ohledu na aktuální nastavení světelné křižovatky.

Odpolední doprava na druhé straně primárně profituje na zmiňovaném odbočovacím pruhu ze severu na východ. V odpoledních hodinách obecně dochází k majoritnímu poptávkovému toku směrem z města, tedy ze severu a částečně z jihu. Tyto toky jsou majoritně obsluhovány právě výpadovkou směrem na východ. Delší interval pro odbočovací pruh ze severu na východ je využíván jednak samotnou vyšší hodnotou, jednak větším podílem na celkové délce cyklu světelné křižovatky.

Pokud by byly spolehlivě statisticky ověřeny doby, ve kterých dochází k proměně charakteru dopravy, vzniknul by prostor pro dvojí nastavení řízení. Vzhledem k mizivé večerní i noční dopravě by tak stačilo měnit pouze ranní a odpolední seřízení světelné křižovatky. Vzniknul by jednoduchý stavový automat s přepínací podmínkou závislou na aktuálním čase. V případě, že není možné logiku křižovatky jakkoliv měnit, je nutné najít optimum napříč oběma dopravními scénáři.

4.4 Srovnání statického nastavení

Předchozí podkapitola pracovala s předpokladem, že nastavení světelného značení lze měnit v čase. Přestože se jednalo pouze o jednodenní změnu logiky systému, ani tento úkon nemusí obecně být možný. Z toho důvodu byla vytvořena tabulka níže, která shrnuje výsledky jednotlivých nastavení nejenom v časové okně, ve kterém byly optimalizovány. Rovněž popisuje výsledky v nastavení opačném, díky čemuž dává náhled do výkonů případného statického nastavení světelného značení. Do srovnání opět vstupují optima nalezená genetickým algoritmem.

Tab. 2: Analýza výsledků optimalizovaných délek front v obou dopravních scénářích.

	Nastaveni	Ranni doprava		Odpoledni doprava	
		avg wait [s]	max wait [s]	awg wait [s]	max wait [s]
Ranni optimum	15, 3, 16	26,1	96	48,29	237
	30, 5, 20	23,05	116	38,65	140
	15, 5, 16	23,57	103	42,84	189
	24, 5, 16	22,93	118	33,30	117
Odpoledni optimum	15, 12, 22	35,16	145	27,53	90
	16, 12, 18	32,31	145	26,93	96
	16, 11, 18	30,13	136	26	100

Z tabulky je patrné výrazné zhoršení obou výsledků algoritmů při řízení světelného značení ve scénáři, na který nebyly optimalizovány. Nyní je nutné zvolit metriku, pomocí které budou u jednotlivého nastavení brány v potaz výsledky z obou scénářů. Pokud by byla k dispozici detailní analýza dopravních toků křižovatkou, bylo by možné z ní vyčíst poměrné zastoupení jednotlivých scénářů v čase. Pomocí znalosti zastoupení by bylo vhodné váženým průměrováním spojit výsledky ranního a odpoledního dopravního scénáře. Jelikož tyto data nejsou k dispozici, k finální analýze je použito aritmetického průměru, předpokládáme tedy uniformní zastoupení ranní a odpolední dopravy v rámci dne. Nejlépe z tohoto srovnání vychází stavy:

- [24, 5, 16] s výsledky: avg wait time: 28,12; max wait time: 117,5
- [16, 11, 18] s výsledky: avg wait time: 28,07; max wait time: 118

Vzhledem k trendu vytváření posloupnosti zelených uvnitř sítě městských světelných křižovatek by tak optimální řešení bylo zvoleno podle nastavení křižovatek navazujících. Obecně lze však říct, že příliš dlouhé trvání vyklizovací šipky, reprezentovanou druhou hodnotou vektoru nastavení, by mohlo být v případě absence odbočovacích aut značně frustrující. Docházelo by totiž k téměř úplnému zastavení dopravy, pokud by zrovna žádná auta neodbočovala. Z tohoto důvodu je zvoleno jako optimální nastavení statického světelného značení posloupnost [24, 5, 16], které byla nalezena jako optimální nastavení v ranním schématu.

5 ZÁVĚR

První část práce se zabývá zavedením terminologie v oblasti řízení světelných křižovatek. Dochází k unifikaci pojmů a vysvětlení několika dále rozváděných úvah. Primární je úvaha o zařazení jednotlivých fází uvnitř cyklu světelné křižovatky. Ta reflektuje poptávkový tok, který má majoritní vliv na vznik dopravních špiček. Dále je provedena rešerše metod obvykle používaných pro optimální řízení světelných křižovatek. V neposlední řadě jsou srovnány různé simulační softwary. Po zvážení faktorů, jakými jsou přístup k licencování, modularita či popularita mezi nově vytvářenými pracemi, byl vybrán software SUMO od společnosti Eclipse.

Pro otestování samotných metod a algoritmů je vitální vytvoření modelu zmíněné světelné křižovatky. Čistě teoretické modely mohou trpět vadami jakými jsou příliš líbivé zobrazení či absence slabin vyplývajících z geografické polohy křižovatky. Vzorem pro práci se tak stala reálná křižovatka nacházející se ve městě Uherské Hradiště. Zvolena byla primárně z důvodu napojení na přilehlý obchvat. Jedná se tedy o dopravní uzel, který má smysl optimalizovat sám o sobě, nikoli s důrazem na plynulost provozu uvnitř samotné sítě městských křižovatek.

Import základních obrysů křižovatky byl proveden za pomoci nástroje OSM web wizard, který využívá Open Street Maps, tedy volně přístupnou databázi dopravních dat napříč zeměkoulí. Pro účely simulace však bylo třeba výrazné úpravy importovaných dat. Jednak došlo k přepisování logiky přímo ve zdrojových souborech vedených v jazyku XML. Jednak byl použit nástroj Netedit, který v uživatelsky přívětivém prostředí dokáže vykonat většinu zamýšlených úprav. V neposlední řadě byly vytvořeny dva simulační scénáře: ranní a odpolední doprava. Oba obsahují složku charakterizující převládající dopravní toky a složku čistě náhodnou. Celkově tak vznikl racionální model, který věrně aproximuje jak rozložení, tak logiku zmiňované reálné křižovatky.

Řídící algoritmy byly psány v jazyku Python s využitím knihovny *TraCI*, která zapisuje data modelu a spouští simulace. Řízení je prováděno za pomocí tří proměnných, které charakterizují délky tří zelených front světelné křižovatky. Sledované výstupy jsou dva, a to průměrná čekací doba a maximální čekací doba účastníků provozu. Jedná se tedy o trojrozměrný optimalizační problém na vstupu a dvojrozměrný na výstupu. Dochází tak k výběru nedominovaných řešení tvořících Pareto frontu, která obsahuje řešení, z nichž se volí to optimální.

Prvním zvoleným algoritmem byl Grid search, neboli prohledávání v mřížce. Došlo k rozdělení stavového prostoru pomocí uniformní mřížky, která sloužila k výběru zkoumaných řešení. Následně byla spuštěna simulace pro každý takto vybraný stav. Metoda trpí vysokou výpočetní náročností, prezentované výsledky jsou však velmi blízké těm optimálním. Obsluha softwaru ve smyslu čtení a přepisování dat simulace je prováděno pomocí přepisování XML souborů.

Druhým zvoleným algoritmem byl genetický algoritmus, konkrétně NSGA-II. Jedná se o implementaci určenou k optimalizaci vícerozměrných problémů. Vyhodnocení vhodnosti populace v každé generaci je vykonáváno s ohledem na již zmíněná nedominovaná řešení. Prohledávání stavového prostoru je tak výpočetně méně náročné a algoritmus má tendence blížit se v rámci generací optimu i v aplikacích bez možnosti dopředného odhadu výsledku.

Na základě výše definovaného modelu a jednotlivých algoritmů je hledáno optimální nastavení světelné křižovatky. Všechna Pareto řešení jsou evidována i s jejich trojrozměrnou vstupní reprezentací, je tedy možné porovnávat jednotlivá řešení v rámci různých nastavení dopravy. V případě možnosti dvoustavového řízení křižovatky jsou představeny nastavení optimální pro ranní a odpolední dopravu zvlášť. Výrazně se od sebe liší v délce jako celku i v míře zastoupení jednotlivých délek zelených. Pokud by změna stavu nebyla možná, dopravní značení by bylo statické, došlo k nalezení optimálního řešení napříč dopravními schématy.

Hlavní přínos práce spočívá v možném přiblížení se optimálnímu nastavení libovolné světelné křižovatky bez nutnosti adaptivního řízení. Obecně se tedy zmíněné postupy a zásady dají aplikovat bez potřeby přebudování či modernizace aktuálních křižovatek. Podmínkou je přesný model, a to jak z hlediska logiky a uspořádání, tak ve smyslu dopravní zátěže.

Doporučením pro navázání na tuto práce je aplikace získaných optimálních délek světelných front uvnitř cyklu světelné křižovatky ve stavovém řízení předlohy pro model křižovatky. Ladění by spočívalo v detailní statistice dopravy v časově významném okně, která by poskytla výborný základ pro volbu reálně optimálního řešení.

6 SEZNAM POUŽITÉ LITERATURY

[1] Eom, Myungeun, and Byung-In Kim. "The traffic signal control problem for intersections: a review.“ *European transport research review* 12.1 (2020): 1-20.

[2] Kochenderfer, Mykel J., and Tim A. Wheeler. *Algorithms for optimization*. Mit Press, 2019. ISBN: 978-02-62039-42-0.

[3] Qadri, Syed Shah Sultan Mohiuddin, Mahmut Ali Gökçe, and Erdinç Öner. "State-of-art review of traffic signal control methods: challenges and opportunities." *European transport research review 12.1 (2020): 1-23.*

[4] Ratrout, Nedal T., and Syed Masiur Rahman. "A comparative analysis of currently used microscopic and macroscopic traffic simulation software." *The Arabian Journal for Science and Engineering* 34.1B (2009): 121-133.

[5] Paz, Alexander, Victor Molano, and Carlos Gaviria. "Calibration of CORSIM models considering all model parameters simultaneously." *2012 15th International IEEE Conference on Intelligent Transportation Systems*. IEEE, 2012.

[6] Horni, Andreas, Kai Nagel, and Kay W. Axhausen. "Introducing matsim." *The multi-agent transport simulation MATSim*. Ubiquity Press, 2016. 3-7.

[7] Zenina, Nadezda, and Yuri Merkuryev. "The Basis for the Transportation Microscopic Simulation Model Validation." *2019 60th International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS)*. IEEE, 2019.

[8] Reza, Imran, Nedal T. Ratrout, and Syed Masiur Rahman. "Calibration protocol for paramics microscopic traffic simulation model: application of neuro-fuzzy approach." *Canadian Journal of Civil Engineering* 43.4 (2016): 361-368.

[9] Jayasooriya, Nadika, and Saman Bandara. "Calibrating and validating VISSIM microscopic simulation software for the context of Sri Lanka." *2018 Moratuwa Engineering Research Conference (MERCon)*. IEEE, 2018.

[10] Biurrun-Quel, Carlos, Luis Serrano-Arriezu, and Cristina Olaverri-Monreal. "Microscopic driver-centric simulator: Linking Unity3d and SUMO." *World Conference on Information Systems and Technologies*. Springer, Cham, 2017.

[11] Nagel, Kai, Richard L. Beckman, and Christopher L. Barrett. "TRANSIMS for transportation planning." *Unifying Themes in Complex Systems*. CRC Press, 2018. 437-444.

[12] Zhao, Dongbin, Yujie Dai, and Zhen Zhang. "Computational intelligence in urban traffic signal control: A survey." *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 42.4 (2011): 485-494.*

[13] Intelligent traffic system. *Alghanim Technologies* [online]. Kuwait: Shopify, c2019 [cit. 2022-05-20]. Dostupné z: https://agtkw.com/project/intelligent-traffic-systems/

[14] Araghi, Sahar, et al. "Influence of meta-heuristic optimization on the performance of adaptive interval type2-fuzzy traffic signal controllers." *Expert Systems with Applications* 71 (2017): 493-503.

[15] Alam, Javed, M. K. Pandey, and Husain Ahmed. "Intellegent Traffic Light Control Systém for Isolated Intersection Using Fuzzy Logic." *Conference on Advances in Communication and Control Systems (CAC2S 2013). Atlantis Press, 2013.*

[16] Herrera, Francisco, Manuel Lozano, and Jose L. Verdegay. "Tuning fuzzy logic controllers by by genetic algorithms." *International Journal of Approximate Reasoning* 12.3-4 (1995): 299-315.

[17] Nguyen, Phuong Thi Mai, Benjamin N. Passow, and Yingjie Yang. "Improving anytime behavior for traffic signal control optimization based on NSGA-II and local search." *2016 International Joint Conference on Neural Networks (IJCNN). IEEE, 2016.*

[18] Yu, Yang, et al. "Optimal reservoir operation using multi-objective evolutionary algorithms for potential estuarine eutrophication control." *Journal of environmental management* 223 (2018): 758-770.

[19] Vidhate, Deepak A., and Parag Kulkarni. "Cooperative multi-agent reinforcement learning models (CMRLM) for intelligent traffic control." *2017 1st International Conference on Intelligent Systems and Information Management (ICISIM)*. IEEE, 2017.

[20] Liang, Xiaoyuan, et al. "A deep reinforcement learning network for traffic light cycle control." *IEEE Transactions on Vehicular Technology* 68.2 (2019): 1243-1253.

[21] Gao, Kaizhou, et al. "Solving traffic signal scheduling problems in heterogeneous traffic network by using meta-heuristics." *IEEE Transactions on Intelligent Transportation Systems* 20.9 (2018): 3272-3282.

[22] Liashchynskyi, Petro, and Pavlo Liashchynskyi. "Grid search, random search, genetic algorithm: A big comparison for NAS." *arXiv preprint arXiv:1912.06059* (2019).

[23] Berrazouane, Mohamed, et al. "Analysis and initial observations on varying penetration rates of automated vehicles in mixed traffic flow utilizing sumo." *2019 IEEE International Conference on Connected Vehicles and Expo (ICCVE)*. IEEE, 2019.

[24] Lopez, Pablo Alvarez, et al. "Microscopic traffic simulation using sumo." *2018 21st international conference on intelligent transportation systems (ITSC)*. IEEE, 2018.

[25] Geografické umístění křižovatky. *Mapy.cz* [online]. 2022 [cit. 2022-05-20]. Dostupné z: https://mapy.cz/zakladni?x=17.4666806&y=49.0487939&z=18&q=uhersk%C3%A9%20hradi%C5%A1t%C4%9B&source=muni&id=3274&ds=2

[26] Amini, Sasan, et al. "Generating and calibrating large-scale, mesoscopic SUMO networks." *SUMO User Conference 2020*. 2020.

[27] L. Bartoš and J. Martolos, *Stanovení intenzit dopravy na pozemních komunikacích: TP 189*, 2. vyd. Plzeň: EDIP, 2012. ISBN 978-80-87394-06-9

[28] Codecá, Lara, et al. "SAGA: An Activity-based Multi-modal Mobility Scenario Generator for SUMO." *SUMO User Conference 2020, Virtual Conference*. 2020.

[29] SUMO simulation files. *Github.com* [online]. San Francisco, California, U.S., 2021 [cit. 2022-05-20]. Dostupné z: https://github.com/ethanpng2021/sumo-example

[30] TraCI User documentation. *Https://sumo.dlr.de/docs* [online]. 2022 [cit. 2022-05-20]. Dostupné z: https://sumo.dlr.de/docs/TraCI.html

[31] Katoch, Sourabh, Sumit Singh Chauhan, and Vijay Kumar. "A review on genetic algorithm: past, present, and future." *Multimedia Tools and Applications* 80.5 (2021): 8091-8126.

[32] Schéma genetického algoritmu. *Github.com* [online]. San Francisco, California, U.S., 2022 [cit. 2022-05-20]. Dostupné z: https://github.com/BozidarVladislav/NSGA-II