ČESKÉ VYSOKÉ
UČENÍ TECHNICKÉ
V PRAZE

F3

Fakulta elektrotechnická
Katedra kybernetiky

Bakalářská práce

Plánování a optimalizace odečtových tras s využitím geografických informací

Dušan Stěhule

Květen 2018

Vedoucí práce: Ing. Milan Rollo, Ph.D.

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Stěhule** Jméno: **Dušan** Osobní číslo: **457225**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra kybernetiky**

Studijní program: **Kybernetika a robotika**

Studijní obor: **Robotika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Plánování a optimalizace odečtových tras s využitím geografických informací

Název bakalářské práce anglicky:

Planning and Optimization of Meter Reading Routes Utilizing the Geographical Information

Pokyny pro vypracování:

1. Seznamte se s problematikou plánování odečtových tras (posloupnost odběrných míst elektrické energie), omezeními vyplývajícími z jejich geografického rozložení a časových omezení pro jednotlivé odečty.
2. Seznamte se s problémem "Obchodního cestujícího - TSP" a jeho rozšířením na více cestujících - MTSP.
3. Seznamte se s problematikou shlukování pro alokaci cílů více jednotkám.
4. Navrhněte možné kombinace výše uvedených přístupů k řešení problému alokace inspekčních míst.
5. Implementujte navržený problém ve Vámi zvoleném programovacím jazyku.
6. Experimentálně ověřte vlastnosti vyvinutého algoritmu na reálných datech a porovnejte se stávajícím řešením.

Seznam doporučené literatury:

[1] Xu, Xiaolong & Yuan, Hao & Liptrott, Mark & Trovati, Marcello.: Two phase heuristic algorithm for the multiple-travelling salesman problem. Soft Computing, 2017.
[2] Tolga Bektas: The multiple traveling salesman problem: an overview of formulations and solution procedures, In Omega, Volume 34, Issue 3, Pages 209-219, ISSN 0305-0483, 2006.
[3] Yupo Chan, S.F. Baker.: The multiple depot, multiple traveling salesmen facility-location problem: Vehicle range, service frequency, and heuristic implementations, In Mathematical and Computer Modelling, Volume 41, Issues 8-9, Pages 1035-1053, 2005.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Milan Rollo, Ph.D., centrum umělé inteligence FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **05.01.2018** Termín odevzdání bakalářské práce: **25.05.2018**

Platnost zadání bakalářské práce: **30.09.2019**

Ing. Milan Rollo, Ph.D.	doc. Ing. Tomáš Svoboda, Ph.D.	prof. Ing. Pavel Ripka, CSc.
podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. Datum převzetí zadání

Podpis studenta

Poděkování / Prohlášení

Chtěl bych poděkovat vedoucímu bakalářské práce Ing. Milanovi Rollovi, Ph.D. za pevné nervy a všechny cenné rady s tvorbou této práce. Další velký dík patří konzultantovi Ing. Petrovi Zajícovi a jeho kolegovi Ing. Janu Pelikánovi za poskytnutá data a rady z praxe. Nesmím opomenout poděkovat ani svoji rodině a přítelkyni, kteří mi byli morální podporou a tvořili mi pevné zázemí při vypracovávání této práce.

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

V Praze dne 25.5.2018

.....

Abstrakt / Abstract

Cílem této práce je seznámit se s algoritmy řešícími problematiku plánování odečtových tras, omezení vyplývající z jejich geografického rozložení a časová omezení pro jednotlivé odečty. Dále se seznámit s algoritmy pro shlukování a alokaci cílů více jednotkám, s problémem obchodního cestujícího - TSP a jeho rozšířením na více obchodních cestujících - MTSP. V praktické části je za úkol z výše uvedených algoritmů navrhnout řešení problému alokace inspekčních míst a navržené řešení implementovat.

Klíčová slova: TSP, MTSP, K-means, Genetický algoritmus, Neuronová síť, Metoda větví a mezi, Hladový algoritmus, Simulované žíhání

The aim of this bachalor thesis is to study algorithms solving problem of planning meter reading routes, limitation resulting from their geographical location and times limitation for individual readings. Then study algorithms for clustering and targeting multiple units, with traveling salesman problem and his extension to multiple traveling salesman problem. In the practical part is aim from this algorithms suggest a solution of problem of clustering inspection posts and implement the designed solution.

Keywords: TSP, MTSP, K-means, Genetic algorithm, Neural network, Branch and bound, Greedy search, Simulated annealing

Title translation: Bachelor thesis (Planning and Optimization of Meter Reading Routes Utilizing the Geographical Information)

Obsah /

Tabulky /

Kapitola 1
Úvod

Cílem této bakalářské práce je automatizovat a optimalizovat plánování odečtových tras společnosti ČEZ Distribuce a.s. Tato společnost má mimo jiné za úkol provádět pravidelné odečty elektroměrů pro účely periodické fakturace u všech svých klientů, a to každý elektroměr jednou za jeden kalendářní rok. Tuto činnost provádějí výkonní zaměstnanci ČEZ Distribuce a.s. Dále společnost zajišťuje tuto činnost na všech odběrných místech na hladině nízkého napětí na celém distribučním území v rozsahu více než 3,5 milionu elektroměrů. Práci v terénu zajišťuje celkem 131 výkonných pracovníků.

V současné době jsou odečtové trasy výkonných pracovníků historicky dané, povětšinou převzaté od regionálních distribučních společností (VČE, STE, SME apod.), které se o daná odběrná místo starala dříve. Neexistuje žádný sofistikovaný software, který by přesně definoval nejefektivnější posloupnost odběrných míst, podle které by výkonní pracovníci měli postupovat. Aktuální zařazení odběrných míst do odečtových tras je založeno na lokální znalosti prostředí výkonných pracovníků a jejich vedení. Vyvstává však otázka, jak moc je tento systém efektivní a jestli by se nedal správnými zásahy zefektivnit. Není známo, jak moc byly dané lokální systémy efektivní, než se začlenily pod správu ČEZ Distribuce a.s. Tato společnost je sice dolaďuje malými zásahy, ale není jasné, zda by se systém nedal razantně zefektivnit zásahy většími.

Dalším problémem současného systému je neustálá fluktuace zákazníků (vznikají nová odběrná místa a zanikají stará, což generuje potřebu správného přiřazení zejména nových odběrných míst do příslušné odečtové trasy), ale i vlastních výkonných zaměstnanců, s jejichž odchodem se ztrácejí cenné znalosti a zkušenosti. Může se tedy stát, že zejména v oblastech s rychlým rozvojem zástavby je historický drobně upravovaný systém již zastaralý a naprosto nevyhovující aktuálním požadavkům. I proto je zde potřeba softwaru, který bude toto všechno zohledňovat a dokáže najít nejoptimálnější řešení (posloupnost odběrných míst pro každého výkonného pracovníka) za všech situací. Softwaru, který bude pracovat v reálném čase a bude tedy schopný reagovat na nenadálé situace (např. onemocnění zaměstnance) a zohlednit je v řešení.

Matematicky lze tuto situaci popsat jako neorientovaný ohodnocený graf, který má zhruba 3.5 milionu vrcholů. Každý vrchol grafu reprezentuje jedno konkrétní odběrné místo. Hrany grafu jsou cesty (silnice, dálnice, chodníky,..) mezi jednotlivými odběrnými místy a jsou ohodnocené délkou cesty (vzdáleností mezi dvěma odběrnými místy). Problém plánování trasy pro všech přibližně 80 odečítačů lze popsat jako Problém více obchodních cestujících (= Multiple traveling salesman problem, dále jen MTSP). Obchodní cestující zde reprezentují jednotlivé odečítače a města reprezentují odběrná místa. Řešením tohoto problému je 80 posloupností měst (pro každého obchodního cestujícího jedna) takových, aby byly z hlediska času realizace odečtu elektroměrů stejně dlouhé a zároveň co nejkratší. V originálním MTSP vycházejí obchodní cestující z jednoho místa a pokryjí celý graf při jedné cestě. V tomto případě však budou vycházet z různých míst a to každý pracovní den. Je možné znázornit tuto situaci tak, že se všechna odběrná místa rozdělí do n skupin (n = počet pracovních dnů) a každou jednu skupinu odběrných míst musí všichni odečítači dohromady navštívit za jeden den.

1.1 Pravidla pravidelných odečtů a rozdělení elektroměrů

Odečet každého konkrétního elektroměru by měl být prováděn přibližně jednou za kalendářní rok. Při plánování je dále důležité zohlednit geografickou polohu a typ odběrného místa. Ta, která mají z nějakého důvodu horší přístupnost, se přednostně řadí do letních měsíců. Horší přístupností je myšlena například vysoká nadmořská výška, kde v zimních měsících bývá hodně sněhu, nebo i odlehlá místa, kde je v zimě riziko náledí na silnicích, a je tedy vhodnější se v těchto měsících pohybovat po městech, kde s největší pravděpodobností budou omezení dopravy minimální. Dalším problémem je přístupnost elektroměrů. Některé elektroměry jsou volně přístupné z veřejného prostranství, což je pro výkonné pracovníky ideální stav. Jiné jsou však umístěny na soukromém pozemku (za plotem), či jsou umístěné přímo v objektu. V takovém případě si výkonný pracovník nejdříve musí domluvit s majitelem objektu jeho součinnost pro zpřístupnění konkrétního elektroměru. Kromě tohoto rozdělení je každý elektroměr zařazen do tzv. „odečtové jednotky“, což je skupina elektroměrů, kterou výkonný pracovník zpracovává zpravidla 1 kalendářní týden.

1.2 Poskytnutá data

Společnost ČEZ Distribuce a.s. poskytla za účelem vypracování této bakalářské práce následující data.

Odečtové jednotky:

kategorie	příklad	popis
Region	Morava	
Týden	34	číslo týdne v daném roce
Měsíc	8	číslo měsíce v daném roce
Číslo OJ	52305341	
Název OJ	OV-Ocelářská, Okružní+Ostravice	
Pracnost OJ 2017 (hod)	27.2	časová náročnost OJ
Počet OM 2017	481	počet elektroměrů
Počet připojených objektů 2017 (ks)	364	počet připojených budov

Tabulka 1.1. Poskytnutá data o odečtových jednotkách

Obsah odečtových jednotek:

kategorie	příklad	popis
Kód místa	53740011-170102	číslo odečtové jednotky - číslo místa
Navrhované pořadí	501	navrhované pořadí v rámci odečtové jednotky
Datum odečtu	23.08.2017 15:33:41	datum a čas posledního odečtu
GPS souřadnice X	49,534930	
GPS souřadnice Y	18,391120	
Město	Ostrava	
Místní část	Vítkovice	
Ulice	Ocelářská	
Č.p.	957	číslo popisné
Č.o.	964	číslo orientační
VGK MM - staré	NA FASÁDĚ	fyzické umístění rozvaděče
VGK OM - nové	RODO	typ odběrného místa (RODO = rodinný dům / BYT)
Přístupnost - nová	A	přístupnost z veřejného místa (A = ano / N = ne)

Tabulka 1.2. Poskytnutá data v rámci odečtových jednotek

Kapitola 2
Problém jednoho a více obchodních cestujících

2.1 Problém obchodního cestujícího

2.1.1 Definice

Problém obchodního cestujícího (anglicky Traveling Salesman Problem - TSP) je diskrétní optimalizační problém, definovaný jako problém nalezení nejkratší hamiltonovské kružnice v úplném neoreintovaném ohodnoceném grafu.

Úplný neorientovaný ohodnocený graf je graf, mezi jehož každými dvěma uzly existuje právě 1 hrana bez dané orientace ohodnocena konkrétní cenou.

Hamiltonovská kružnice je cyklus, který obsahuje každý vrchol grafu právě jedenkrát.

Laicky by se dalo říct, že se jedná o problém, kdy je dáno n měst, mezi každýma dvěma existuje silnice o známé délce, a úkolem je najít nekjratší možnou cestu, která prochází všemi městy a vrací se zpět do výchozího města. [1]

2.1.2 Obecné řešení a jeho složitost

Problém obchodního cestujícího patří mezi nedeterministicky polynomiálně- těžké (NP-těžké) úlohy. To znamená, že neexistuje algoritmus, který by pro obecný vstup našel řešení v reálném čase. Neví se ani, zda takový algoritmus může vůbec existovat. Existuje však algoritmus, který dokáže v polynomiálním čase ověřit správnost výsledku.

Existují mimo jiné dvě skupiny výpočetních úloh: polynomiální (P) s časovou složitostí n^k (n = počet prvků, k = konstanta) a exponenciální (E) s časovou složitostí k^n. Třída NP se časovou složitostí nachází mezi P a E. Úkolem jednoho (ze sedmi) problémů tisíceletí je dokázat, či vyvrátit, že se třídy P a NP rovnají. [2]

2.1.3 Algoritmy řešící problém obchodního cestujícího

2.1.4 Hrubá síla

Principielně velmi jednoduchý algoritmus, při kterém se vygenerují všechny možné cesty (permutace měst), spočítá se jejich cena (suma vzdáleností mezi městy) a vybere se cesta s nejnižší cenou. Velkou výhodou je, že tento algoritmus jako jediný najde vždy optimální řešení. Je ovšem použitelný pouze pro malý rozsah vzorků, maximálně v řádu desítek až stovek.

Je dáno n měst. V prvním (=výchozím) městě existuje (n-1) možností, jaké město navštívit dále. Ve druhém městě existuje opět (n-1) cest k ostatním městům, ale z těchto (n-1) měst bylo již jedno navštíveno, zbývá tedy (n-2) možností. Ve třetím městě

to bude (n-3) možností atd. V předposledním městě zbývá už jen jedno nenavštívené město.

Po vynásobení všech možnosti z jednotlivých kroků, vyjde celkový počet různých cest:

Celkový počet cest = (n-1)(n-2)...1 = (n-1)!

V tabulce 2.1. je naznačen vývoj počtu možných cest (řešení) v závislosti na počtu měst. Jak je vidět, počet řešení se zvyšuje faktoriálně, proto se tento algoritmus hodí pouze pro malé vzorky dat.

počet měst	počet cest
2	1
5	24
20	$1.216 * 10^{17}$
50	$6.082 * 10^{62}$
200	$3.943 * 10^{372}$

Tabulka 2.1. Počet cest v závislosti na počtu měst

2.1.5 Heuristika

Algoritmy využívající heuristiku nenajdou vždy optimální řešení, avšak s vhodným nastavením heuristiky jsou schopny najít řešení velmi dobré, blížící se tomu optimálnímu. Jsou použitelné i pro větší počet prvků, díky podstatně menší časové náročnosti než u algoritmů založených na hrubé síle.

2.2 Problém více obchodních cestujících

Jako problém více obchodních cestujících (anglicky Multiple traveling salesman problem - MTSP) lze označit situaci, kdy dva, tři nebo více obchodních cestující typicky startujících v jednom městě začne procházet všechna ostatní města, přičemž je třeba, aby každé město bylo navštíveno právě jedním obchodním cestujícím a to právě jednou. Řešením tohoto problému je taková optimalizovaná cesta, aby celková vzdálenost absolvovaná všemi obchodními cestujícími byla co nejkratší, respektive aby doba cesty nejpomalejšího obchodního cestujícího byla co nejkratší. [3]

Existují dvě základní strategie řešení:
1. Každý obchodní cestující navštíví stejný počet měst.
2. Každý obchodní cestující urazí stejnou vzdálenost mezi městy.

2.2.1 Definice

S m obchodními cestujícími a n městy, je MTSP definován jako komplexní graf

(1) $G = G(V, E)$,

kde $V = [v_1, ..., v_n]$ reprezentuje města s počátečním umístěním ve vrcholu v_0. Každá hrana (v_i, v_j) je charakterizována váhou d_{ij} $(d_{ij} > 0, d_{ii} = \infty, v_i, v_j \in V)$, která představuje cenu cesty (z hlediska vzdálenosti nebo času) mezi městy i a j. [3]

Dále je definován maximální počet měst Q pro každého obchodního cestujícího dopomáhající dosažení rovnováhy pracovního zatížení mezi jednotlivými obchodními cestujícími.

(2) $Q = \frac{n}{m}$

Z toho vyplývá, že jedním z doprovodných cílů MTSP je určit m sekvenci Hamiltonovských cyklů nad G s celkovou nejnižší cenou tak, aby všichni obchodní cestující dohromady navštívili každé město právě jednou. Objektivní funkce MTSP je tedy:

(3) $minZ = \sum_{t=0}^{m} \sum_{i=0}^{n} \sum_{j=0}^{n} r_{ijt} d_{ij}$

(4) $\sum_{t=0}^{m} r_{ti} = 1 \; i = 1, ..., n$
(5) $\sum_{i=0}^{n} r_{ijt} = y_{tj} \; j = 1, ..., n \; \forall t$
(6) $\sum_{i=0}^{n} r_{ijt} = y_{ti} \; i = 1, ..., n \; \forall t$
(7) $\sum_{i=0}^{n} y_{ti} \leq Q \; t = 1, ..., m,$

kde

i odkazuje na město ($i = 1, .., n$), kde startovní místo je rovno 0,
t odkazuje na obchodního cestujícího ($t = 1, ..., m$), kde m je celkový počet obchodních cestujících,
d_{ij} reprezentuje vzdálenost mezi městy i a j,
Q je maximální počet měst, které může navštívit jeden obchodní cestující,
$r_{ijt} \in [0, 1]$; $r_{ijt} = 1$ když obchodní cestující t cestuje přímo z města i do města j, pokud ne, tak $r_{ijt} = 0$,
$y_{ti} \in [0, 1]$; $y_{ti} = 1$ když obchodní cestující t navštíví město i, pokud ne, tak $y_{ti} = 0$.

Rovnice 4 říká, že každé město musí být navštívené právě jedním obchodním cestujícm. Rovnice 5 a 6 říkají, že všechny obchůzky každého obchodního cestujícího musí začínat a končit ve stejném městě. A nakonec, rovnice 7 říká, že počet navštívených měst konkrétním obchodním cestujícím nesmí překročit konkrétní hodnotu (maximální počet měst Q). [3]

Kapitola 3
Řešení problému jednoho a více obchodních cestujících

3.1 Dvoufázový heuristický algoritmus pro MTSP

Dvoufázový heuristický algoritmus (anglicky Two phase heuristic algorithm - TPHA) se skládá ze dvou částí (algoritmů): Clusteringového (např. K-means), jehož cílem je rozdělit všechna individuální města do m skupin, v rámci kterých již funguje heuristický (např. genetický) algoritmus hledající nejkratší cestu. [3]

3.2 Rozdělení měst do skupin pomocí vylepšeného K-means

V originálním K-means algoritmu jsou náhodně vybrána města reprezentující středy (centroidy) odpovídajících skupin (klastrů). Následně jsou pomocí fitnes funkce (a nebo pomocí některé jiné selektivní funkce) k těmto centroidům přiřazena blízká města. Tento postup však neumožňuje splnit cíl vyváženého počtu měst. [3]

Nechť $V = \{V_i : i = 1, ..., n\}$ je množina n měst a počáteční skupina klastrů $s = (c_1, ..., c_k), \vec{v}_j$ má c_j jako centroid, kde $\vec{v}_j = v_{ij}, j = 1, ..., k$. Počet měst v každé skupině q_j je roven 0.

Vylepšený K-means algoritmus zahrnuje tyto kroky:

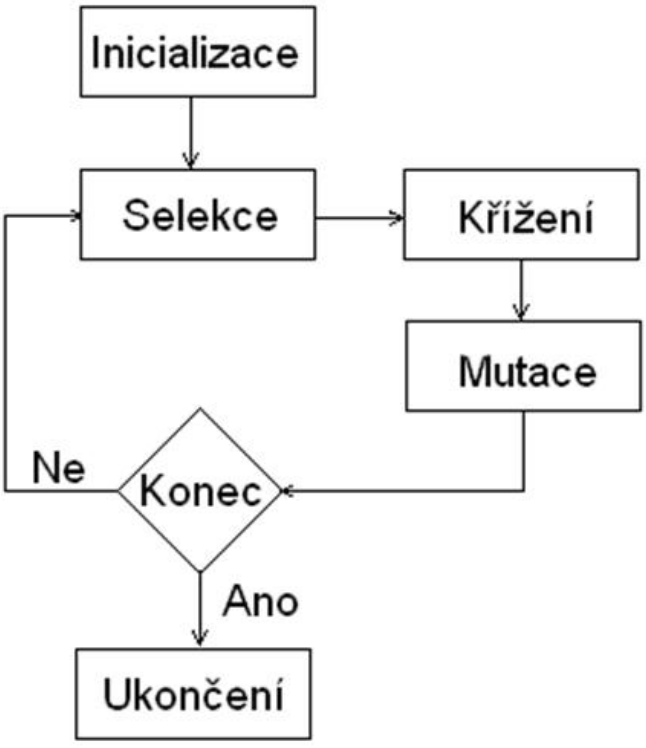
1 Nastavení kapacity klastru, kde $Q = \frac{n}{m}$ je maximální kapacita.
2 Vypočtení vzdálenosti každého města $v_i \in V$ od centrálního klastru $\vec{v}_j$ dle vztahu:
$||v_i - \vec{v}_j|| = \sqrt{(x_i - \vec{x}_j)^2 + (y_i - \vec{y}_j)^2}, i = 1, ..., n, j = 1, ..., k,$
a seřazení vypočtených vzdáleností $||v_i - \vec{v}_j||$ vzestupně dle velikosti.
3 Pokud existuje město $v_i \in V$ s minimální vzdáleností od centroidu c_j, pak $q_j = q_j + 1$. Pokud je $q_j < Q$, zbývá přiřadit město v_i do c_i. V opačném případě je hodnota vzdálenosti $||v_i - \vec{v}_j|| = \infty$. Tento bod se opakuje, dokud nejsou všechna města přiřazena k některé skupině.
4 Souřadnice centroidu se aktualizují
$\vec{x}_j = \frac{1}{|c_j|} \sum_{v_i \in c_j} x_i$
$\vec{y}_j = \frac{1}{|c_j|} \sum_{v_i \in c_j} y_i,$
kde $|c_j|$ je počet měst ve skupině c_j. Pokud se souřadnice centroidu nemění, algoritmus pokračuje bodem 5), v opačném případě bodem 2).
5 Proces shlukování je kompletní s výstupním klastrem $s = (c_1, ..., c_k)$. [3]

3.3 Plánování trasy pomocí genetického algoritmu

V rámci každé skupiny (viz. bod 3.2 Dvoufázový heuristický algoritmus pro MTSP) se aplikuje genetický algoritmus patřící do třídy evolučních algoritmů. Vychází z Darwinovy teorie přirozeného výběru. V přírodě existují populace živočišných druhů složené z různých jedinců. Konkrétního jedince je možné si představit jako soubor genů, kde se jednotlivé geny shlukují do větších celků, chromozómů. Při vzniku nového jedince dochází k přebírání genů od obou rodičů. Geny se tedy mísí- dochází ke křížení. Výjimečně může dojít i k náhodné změně některých z genů- mutaci. Nově vzniklý jedinec tedy kombinuje geny dvou jedinců minulé generace (křížení), nebo může obsahovat geny úplně nové (mutace). Jelikož je jedinec se špatnými geny znevýhodněn oproti jedincům s dobrými geny (brzy zemře, nenajde vhodného partnera k páření apod), špatné geny se postupně eliminují a v nových generacích se nacházejí jedinci s lepšími geny než v generacích starších. [4]

Analogicky k tomu při řešení MTSP za použití genetického algoritmu si lze pod pojmem jedinec představit konkrétní řešení (v rámci dané skupiny), tedy konkrétní pořadí měst. Pomocí fitness funkce se tomuto jedinci přiřadí hodnota vyjadřující, o jak moc dobré řešení se jedná. Pomocí různých strategií výběru se vybere několik řešení, ze kterých se vytvoří nová populace. Přitom jsou různě křížena (průměrována, jsou zaměňovány jednotlivé části řešení), nebo náhodně pozměněna. Vzniknou tedy noví jedinci (řešení), kteří mohou být lepší jak ti předchozí. Tento algoritmus se opakuje do doby, dokud není řešení dostatečně dobré. [5]

Schéma genetického algoritmu je znázorněno na obrázku 3.1. Schéma genetického algoritmu.



Obrázek 3.1. Schéma genetického algoritmu

parametr	popis
G	celkový počet iterací
P_{size}	velikost populace
p_c	pravděpodobnost křížení
p_m	pravděpodobnost mutace
X^k	populace k-té generace
X_i^k	i-tý jedinec v populaci k-té generace
D_i^k	hodnocení i-tého jedince v X^k, $D_i^k = D(X_i^k)$
F_i^k	hodnocení i-tého jedince v X^k, $F_i^k = F(X_i^k)$
$[subx_1 subx_2] = CS(x_1, x_2)$	nová generace sub x_1 a sub x_2, která vznikla mutací nebo křížením rodičů x_1 a x_2
U (0,1)	funkce náhodně generující hodnoty od 0 do 1 (rovnoměrné rozdělení)
X_{best}	aktuální nejlepší hodnota
D_{best}	aktuální nejlepší hodnota

Tabulka 3.1. Parametry genetického algoritmu

3.3.1 Fitness funkce

Fitness funkce se používá k ohodnocení jednotlivých jedinců v dané generaci. Je stěžejním krokem genetického algoritmu, který je díky ní použitelný i pro velké soubory dat. Je definována jako

$$F(x) = \frac{1}{D(x)},$$

kde x označuje konkrétního jedince a D(x) značí součet vzdáleností mezi městy, které urazí všichni obchodní cestující v rámci tohoto řešení. Vyšší hodnota fitness funkce znamená, že jedinec bude spíše vybrán do příští generace. [6]

3.3.2 Strategie výběru

Spolu s fitness funkcí je dobrá strategie výběru dalším důležitým předpokladem pro vysokou efektivitu genetického algoritmu.

Existují různé strategie výběru:

3.3.3 Metoda ruletového kola

Aby funkce neuvázla v lokálním extrému, ale našla skutečně optimální řešení, je třeba přidat do rozhodovací fáze prvek náhody. Tato metoda se podobá klasickému ruletovému kolu, jen s tím rozdílem, že vhodnější jedinci jsou zvýhodňováni před těmi méně vhodnými a to tak, že na ruletovém kole zabírají větší výseč. K vytvoření takového ruletového kola je potřeba sečíst fitness funkci všech jedinců v populaci a poté proporcionálně přiřadit každému jedinci odpovídající kruhovou výseč. [6]

3.3.4 Elitismus (elitářská strategie)

Při funkci genetického algoritmu nelze vyloučit možnost, že i jedinec s velmi vysokou fitness funkcí nebude vlivem náhodných jevů vybrán. Funkcí elitářské strategie je tedy zapamatovat si několik nejlepších jedinců a přímo je přenést do nově vytvořené populace. Tím se zajistí, že i v nepříznivých případech křížení a mutací (nově vzniklí jedinci budou horší, než ti předchozí) bude zachována stále stejná úroveň kvality jedinců. [6]

3.3.5 Turnaj

Při této metodě výběru se náhodně vybere N jedinců a do další generace postoupí ten nejlepší z nich. Tento proces se opakuje stále dokola, dokud není naplněn očekávaný počet jedinců v příští generaci. Tato metoda zahrnuje jak prvek náhody, tak i respektuje kvalitu řešení. [6]

Následuje podrobně popsaný příklad algoritmu, který kombinuje dvě metody a sice metodu ruletového kola a elitářskou strategii.

1 Je dána velikost populace P_{size} a i-tý jedinec X^k a sice X_i^k. Po vyhodnocení fitness funkce a následném seřazení výsledků v sestupném pořadí se jedinci na prvních místech přesunou do další generace (elitismus).
2 Pravděpodobnost výběru každého jedince je dána jako

$$p_i = \frac{F_i^k}{\sum_{i=1}^{G} F_i^k},$$

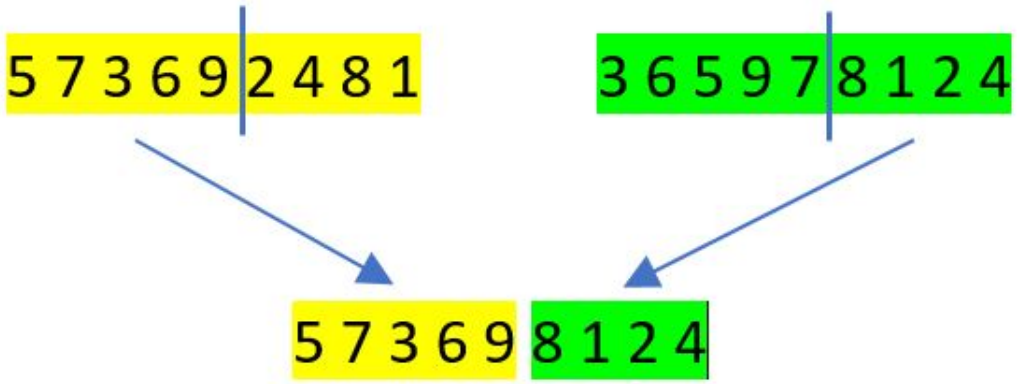
kde p_i je přiřazeno každému jedinci X_i^k na základě provedeného výpočtu (metoda ruletového kola).
3 Příští generace se vybírá pomocí metody ruletového kola, kdy se náhodně vygeneruje jednotně distribuované číslo δ z intervalu (0,1). Pokud platí rovnost

$$\sum_{j=0}^{i-1} p_j \leq \delta \leq \sum_{j=0}^{i} p_j,$$

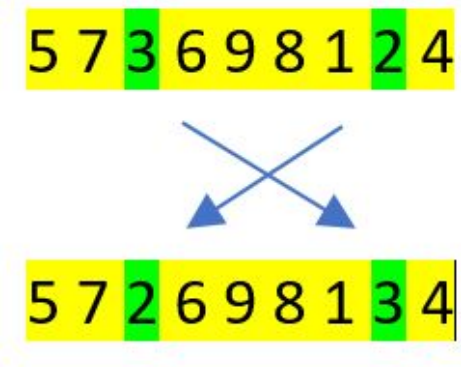
pak je jedinec X_i^k vybrán do další generace. Postup se opakuje pro všechny jedince (řešení). [3]

3.3.6 Křížení

Dva rodičovské chromozomy P_1 a P_2 jsou vybrány na základě pravděpodobnosti p_c. Vzniknou dvě části ΔP_1 a ΔP_2. Nově vzniklému jedinci je přiřazena část ΔP_1, zatímco ekvivalentní část chromozomu P_2 je ignorována. Druhá část chromozomu P_2 doplní zbývající místo v chromozomu potomka, které má tak kompletní chromozom vzniklý křížením těch rodičovských. [3]



Obrázek 3.2. Schéma křížení



Obrázek 3.3. Schéma mutace

3.3.7 Mutace

Mutace hraje důležitou roli při vylepšování daných jedinců při zachování variability obyvatelstva. Zabraňuje také uváznutí algoritmu v lokálním maximu. S pravděpodobností p_m jsou dvě města prohozena v plánovaném pořadí návštěvy. [6]

3.4 Metoda větví a mezí

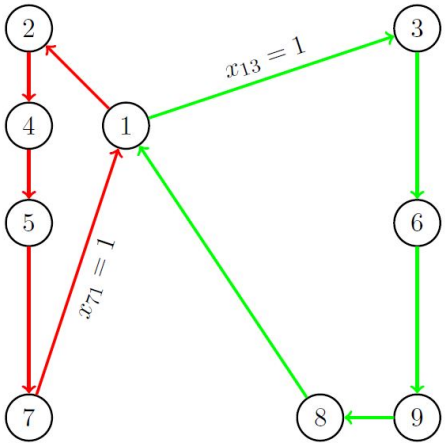
Pro formulaci algoritmu řešení pomocí metody větví a mezí je nutné nejříve převést MTSP na ILP (integer linear program = celočíselný lineární program), čehož je možné dosáhnout pomocí Bektasovi metody. V prvním kroce se očíslují všechny uzly, přičemž depo je uzel číslo 1. Nechť C_{ij} je nejkratší průměrný čas cesty z uzlu i do uzlu j. Nechť ukazatel x_{ij} je 1, pokud je cesta použita v aktuálním řešení, a 0 pokud nikoliv. Na obrázku 3.4. je příklad trasy dvou obchodních cestující. Po aplikování Bektasovy metody lze stav uvedený na obrázku 3.4. vyjádřit následovně: [7]

$$\begin{pmatrix} x_{11} & x_{12} & x_{13} & x_{14} & x_{15} & x_{16} & x_{17} & x_{18} & x_{19} \\ x_{21} & x_{22} & x_{23} & x_{24} & x_{25} & x_{26} & x_{27} & x_{28} & x_{29} \\ x_{31} & x_{32} & x_{33} & x_{34} & x_{35} & x_{36} & x_{37} & x_{38} & x_{39} \\ x_{41} & x_{42} & x_{43} & x_{44} & x_{45} & x_{46} & x_{47} & x_{48} & x_{49} \\ x_{51} & x_{52} & x_{53} & x_{54} & x_{55} & x_{56} & x_{57} & x_{58} & x_{59} \\ x_{61} & x_{62} & x_{63} & x_{64} & x_{65} & x_{66} & x_{67} & x_{68} & x_{69} \\ x_{71} & x_{72} & x_{73} & x_{74} & x_{75} & x_{76} & x_{77} & x_{78} & x_{79} \\ x_{81} & x_{82} & x_{83} & x_{84} & x_{85} & x_{86} & x_{87} & x_{88} & x_{89} \\ x_{91} & x_{92} & x_{93} & x_{94} & x_{95} & x_{96} & x_{97} & x_{98} & x_{99} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Nechť potenciál u_i je počet uzlů, které obchodní cestující dopasavaď navštívil, když přijíždí do uzlu i. V obrázku 3.4. by to bylo takto:

$$u_2 = 1, u_4 = 2, u_5 = 3, u_7 = 4; u_3 = 1, u_6 = 2, u_9 = 3, u_8 = 4$$

Nechť p je maximální počet měst, včetně návratu do depa, které může obchodní cestující navštívit. Hodnota p může být nastavena manuálně tak, aby vyhovovala potřebám konkrétního řešení. Pokud má být například rozděleno 10 měst po 5 městech mezi 2 obchodní cestující, hodnota parametru bude p=6. [7]



Obrázek 3.4. Příklad cesty dvou obchodních cestující [7]

Problém lze vyjádřit pomocí následující rovnice:

$$min \sum_{i=2}^{n} \sum_{j=2}^{n} C_{ij} x_{ij}$$

s podmínkami:

(1) $\sum_{j=2}^{n} x_{1j} = m$
(2) $\sum_{i=2}^{n} x_{i1} = m$
(3) $\sum_{i=1}^{n} x_{ij} = 1, j = 2, ..., n$
(4) $\sum_{j=1}^{n} x_{ij} = 1, i = 2, ..., n$
(5) $u_i - u_j + p x_{ij} <= p - 1, i = 2, ..., n, j = 2, ..., n$
(6) $x_{ij} \in 0, 1, i, j = 1, ..., n$
(7) $u_i \in 1, 2, ..., p - 1, i = 2, ..., n$

1. podmínka říká, že m cest začíná v depu (uzel 1). 2. podmínka říká, že zde také m cest končí. 3. a 4. podmínka dohromady říkají, že všechna ostatní města jsou navštívena právě jednou. Tyto čtyři podmínky nejsou samy o sobě dostačující, umožňují totiž řešení, které obsahují cykly, jenž nejsou propojeny s depem. Proto existuje ještě 5. podmínka, která právě toto řešení zakazuje.

Tento problém (ILP) je řešitelný pomocí metody větví a mezí. Metoda LP relaxace (linear programing relaxation = lineární programování relaxace) zkoumá případ, pokud by neplatily podmínky, že x_{ij} a u_i musí být celočíselné. Místo toho to mohou být reálné hodnoty mezi 0 a 1 pro x_{ij} a mezi 1 a (p-1) pro u_i. Tento případ je řešitelný pomocí duálních simplexních algoritmů. Pokud řešení obsahuje $0 < x_{ij} < 1$, může být rozvětvené do dvou případů $x_{ij} = 0$ a $x_{ij} = 1$. To už však odpovídá výše zmíněné metodě ILP s podmínkou $x_{ij} = 0$ nebo $x_{ij} = 1$. Tyto větve nebo případy jsou uloženy v aktivním listu, ze kterého jsou nepřetržitě brána data pro řešení LP relaxace. Pokud má aktuálně testovaná větev neuskutečnitelnou LP relaxaci, je tato větev vyloučena. Pokud

LP relaxace najde řešení s vyšší cenou, než je cena aktuálně nejlepšího řešení, není tato větev dále zkoumána, protože dle definice jsou náklady u potomka vyšší než náklady u rodiče. Takováto větev je tedy rovněž prořízнuta. Když metoda LP relaxace nalezne celočíselné řešení, je daná větev také prořízнuta, protože je (z výše uvedených důvodů) zbytečné dále pátrat mezi potomky. Algoritmus končí a vrátí řešení v momentě, kde je aktivní list prázdný. [8]

Zbývající otázkou je, jak prohledávat tento strom. Které větve z aktivního listu vzít jako první? Je možné prohledávat strom do šířky. V ideálním případě bude řešení ve vrchní části stromu a nebude třeba prohledávat všechny větve. Další možností je prohledávat graf do hloubky, kde bude v nejlepším případě rychle nalezeno dobré celočíselné řešení a ostatní větve tak mohou být prořezány. Kombinací těchto dvou přístupů vznikne metoda přepínání: nejdříve bude graf prohledáván do hloubky, dokud nebude nalezeno celočíselné řešení, a poté se přepne na prohledávání do šířky. Prohledávání do šířky lze docílit tak, že nové vytvořené větve budou vloženy na konec aktivního listu. Naopak prohledávání do hloubky lze docílit tak, že se nově vzniklé větve přidají na začátek tohoto seznamu. [8]

3.5 Hladový algoritmus

Hladový algoritmus je jednoduchý způsob jak rychle prohledat celý stavový prostor. Bohužel však nedokáže zaručit nalezení optimálního řešení a často uvázne v lokálním extrému. Prostor řešení sestává ze všech možných cest s danou hodnotou n (počet měst), m (počet obchodních cestujících) a p (maximální počet měst, které má obchodní cestující navštívit včetně návratu do depa).

3.5.1 Reprezentace řešení

Řešení je reprezentováno maticí. Každý řádek říká, která města a v jakém pořadí konkrétní obchodní cestující navštíví. Nechť n=4, m=2 a p=4. Matice bude rozměrově vypadat následovně:

$$\begin{pmatrix} 2 & 4 & 0 \\ 3 & 0 & 0 \end{pmatrix}.$$

Takovéto řešení říká, že první obchodní cestující navštíví město 2, poté 4 a nakonec se vrátí zpět do depa. Druhý obchodní cestující navštíví pouze město 3 a vrátí se do depa. Depo je označené číslem 1, avšak neuvádí se v této reprezentaci řešení, jelikož je známé, že všichni obchodní cestující začínají a končí v depu (městě 1). Nuly mohou být ignorovány, jsou zde pouze z důvodu, aby naznačovaly, kolik měst ještě může daný obchodní cestující navštívit. [9] [7]

3.5.2 Sousední řešení

Sousední řešení N daného řešení B je definováno jako jakékoliv řešení, které vznikne z daného řešení B provedením jedné z následujících pěti operací:

1 Intraroute inversion: Vybere se skupina měst jednoho obchodního cestujícího a obrátí se její pořádí
2 Intraroute switching: Vyberou se dvě skupiny měst v rámci měst jednoho obchodního cestujícího, které se vzájemně prohodí.

3 Intraroute instertion: Vybere se město/skupina měst a dá se na jiné místo v pořadí téhož obchodního cestujícího.
4 Interroute switching: Vybere se skupina měst jednoho obchodního cestujícího a skupina měst jiného. Tyto dvě skupiny se pak vzájemně prohodí. Tyto skupiny jsou vybírány tak, aby jakákoliv cesta obchodního cestujícího neobsahovala méně než 1 město, čí více než (p-1) měst.
5 Interroute transfer: Vybere se skupina měst jednoho obchodního cestujícího a zařadí se na konec cesty jiného. Tato skupina a obchodní cestující, kterému je následně přiřazena, jsou vybírány tak, aby jakákoliv cesta obchodního cestujícího neobsahovala méně než 1 město, či více než (p-1) měst. [7]

3.5.3 Popis algoritmu

Cena celkové cesty je součet všech dílčích cest mezi městy s výjimkou cest spojených s výchozím městem.

Základní myšlenkou hladového algoritmu je vybrat náhodné počáteční řešení, porovnat ho s náhodným sousedním řešením a vybrat to levnější z nich. Konec nastává, když algoritmus uvázne v lokálním extrému. Celý proces se opakuje r-krát a výsledné řešení je to nejlevnější řešení ze všech mezivýsledků. Celý proces vypadá následovně:

1 Vybere se náhodné počáteční řešení z množiny všech řešení. Toho lze dosáhnout generováním náhodných permutací vektorů (2,3,..,n), jejich náhodným rozřezáním na m kusů, přičemž každý vzniklý kus nesmí být delší než (p-1) měst. Toto řešení se položí jako nejlepší řešení B. Je-li toto první iterace algoritmu, nastaví se toto řešení také jako výsledné nejlepší řešení O.
2 Zvolí se náhodné sousední řešení N řešení B tak, že se náhodně vybere jedna z pěti výše popsaných operací a provede se náhodným způsobem. Pokud má sousední řešení N nižší cenu než B, B = N. Dále se ověří, zda není sousední řešení N levnější než výsledné řešení O. Pokud ano, O = N.
3 Předchozí dva kroky se opakují, dokud se B nemění v posledních π pokusech. Když k tomu dojde, předpokládá se, že bylo nalezeno lokální optimum. Hodnota parametru trpělivosti π se hledá experimentálně.
4 Opakují se všechny výše uvedené kroky r krát, přičemž r je počet opakování. Jedná se o další parametr, který lze získat experimentálně. [7]

Když proběhl celý algoritmus r krát, vrátí hodnotu nejlepšího celkového řešení O. Ačkoliv je experimentování s hodnotou r možné, obvykle se nastavuje na hodnotu $r = 1$.

3.6 Simulované žíhání

Ačkoliv hladový algoritmus funguje relativně rychle, není považován za sofistikovanou heuristickou metodu. Naproti tomu metoda simulovaného žíhání je sofistikovanější a méně citlivá na zaseknutí v lokálním extrému. Její koncept je založen na volnosti pohybu částic v kovu. Když je kov horký a roztavený, částice se mohou pohybovat volně, když však kov začíná chladnout a tuhnout, částice jsou stále více vázány na konkrétním místě. V kontextu prohledávání stavového prostoru spočívá podobnost v tom, že tato metoda umožňuje nahradit současné řešení sousedním ve velmi malé závislosti na optimu, ale s postupem času ho stále více nutí do lokálního optima. Potenciál metody je ve veliké

volnosti pohybu aktuálního řešení při hledání oblasti globálního minima, přičemž poté se volnost pohybu omezí a najde se přesné optimální řešení. Pravděpodobnost s jakou je přijato dražší řešení odpovídá tomu, jak moc je toto řešení dražší. [7] [10]

3.6.1 Jednoduchý problém

Simulované žíhání prohledává stejný stavový prostor jako hladový algoritmus a je použita stejná definice sousedních řešení, vzniklých pěti různými operacemi popsanými v kapitole 3.5.2. Podrobněji je možné použitý algoritmus popsat následovně:

1 Vybere se náhodné počáteční řešení B podobně jako v hladovém algoritmu a nastaví se jako aktuálně nejlepší řešení O.
2 Teplota T se nastaví na nějakou fixní počáteční teplotu T_0. Parametr počáteční teploty T_0 se určuje experimentálně.
3 Při aktuální teplotě T se zvolí náhodné sousední řešení N řešení B. Pokud má sousední řešení N nižší cenu než B, nastaví se B = N. Pokud má sousední řešení N nižší cenu než aktuálně nejlepší řešení O, nastaví se O = N. Pokud má sousední řešení N vyšší nebo stejnou cenu jako B, nastaví se B = N s pravděpodobností:
$p(N, B, T) = exp(-(cost(N) - cost(B))/T)$
4 Krok 3 se opakuje t krát s danou teplotou T, kde t je parametr určený experimentálně. Poté se nastaví nová teplota $T_{new} = \alpha.T_{old}$. Parametr α mívá běžně hodnotu 0.95, avšak může být nastaven libovolně.
5 Když je teplota T menší než teplota T_{end}, algoritmus se ukončí a vrátí nejlepší řešení O.

U metody simulovaného žíhání je relativně složité nalézt vhodné parametry T_0, T_{end}, t a α. [7] [10]

3.6.2 Souvislost s hladovým algoritmem

Implementace hladového algoritmu lze velmi snadno dosáhnout z implementace simulovaného žíhání. Stačí pouze nastavit pravděpodobnost $p(N, B, T) = 0$, jinými slovy zakázat všechna dražší řešení. Ostatní změny jsou minimální, kritérium ukončení algoritmu by mělo být nezávislé na teplotě a jiné nerelevantní proměné pro hladový algoritmus by neměly být brány v úvahu.

3.7 Neuronová síť

Zcela odlišný způsob řešení optimalizačních problémů přichází s výzkumem umělé inteligence. Algoritmy založené na neuronové síti jsou systémy, jenž se naučí rozeznávat kvalitu řešení, které je reprezentováno jako sada hodnot pro danou proměnnou. Dělají to buď skrze pozorování velkých vzorků dat špatného a dobrého řešení, v tomto případě levného realizovatelného a drahého/nerealizovatelného řešení, nebo pozorováním kvality řešení vzhledem k nějakému měřítku (pravidlům). Algoritmus neuronové sítě řešící MTSP vymyslel pan Wacholder et al., jehož práce staví na algoritmu Hopfieldovi neuronové sítě pro řešení TSP. Wacholder představuje MTSP jako neuronovou síť s náhodným spouštěním napětí na uzlech a číselné schéma, se kterým by počáteční stav měl konvergovat k proveditelnému řešení s nízkými náklady. [11]

3.7.1 Jednoduchý problém

MTSP může být reprezentován pomocí neuronové sítě. Stav této sítě je reprezentován (n + m - 1) x (n + m - 1) maticí V obsahující pouze 1 a 0. Hodnota V_{ki} je rovna 1, právě když je uzel k i-tý uzel, který má být navštíven. Posledních (m-1) sloupců V reprezentuje návrat do depa. Efektivně, (m-1) virtuálních bodů je uvedeno se stejnou vzdáleností k ostatním bodům, jako má depo. Jako takové je depo reprezentováno jako první a (m - 1) posledních sloupců. Řešení

$$\begin{pmatrix} 2 & 4 & 0 \\ 3 & 0 & 0 \end{pmatrix}$$

,

pro (n,m,p) = (4,2,4) odpovídá stavu neuronové sítě

$$\begin{pmatrix} & 1st & 2nd & 3rd & 4th & 5th \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 1 \\ 4 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

,

protože první návštívený je uzel 1, další v pořadí je uzel 2, dále uzel 4, čtvrtá návštěva je v uzlu 5 (depo, konec cesty prvního obchodního cestujícího) a pátá návštěva je v uzlu 3. Nakonec následuje návrat do uzlu 1. [11]

Neuronovou síť je možné považovat za shluk elektricky nabitých neuronů s jasnými pravidly distribuce elektřiny. V uskutečnitelném koncovém stavu neuronové sítě jsou prvky výše uvedené matice přibližně rovny 0 nebo 1. Každý prvek matice představuje jeden neuron neuronové sítě s aktivací mezi 0 a 1, kde aktivovaný neuron má hodnotu blízkou 1. Aktivace neuronu je založena na změně elektrického náboje daného neuronu. Pokud má neuron v místě (k,i) náboj U_{ki}, jeho aktivace je určena sigmoidní funkcí $g : (-\infty, \infty) -> (0, 1)$:

$$O < V_{ki} = g(U_{ki}) = \frac{1}{2}(1 + tanh(\frac{U_{ki}}{u_{00}})) < 1$$

s konstantou u_{00} závislou na daných parametrech. [11]

Wacholder zavádí soubor pěti pravidel pro konečný stav MTSP neuronové sítě vyjádřený následujícími rovnicemi:

$$(8)\ \frac{1}{2}\sum_{k=1}^{n+m-1}\sum_{i=1}^{n+m-1}\sum_{j=1 j\neq i} V_{ki}V_{kj} = 0$$

$$(9)\ \frac{1}{2}\sum_{i=1}^{n+m-1}\sum_{k=1}^{n+m-1}\sum_{l=1,l\neq k} V_{ki}V_{li} = 0$$

$$(10)\ \frac{1}{2}((\sum_{k=1}^{n+m-1}\sum_{i=1}^{n+m-1} V_{ki}) - (n+m-1))^2$$

$$(11)\ \frac{1}{2}\sum_{k=n+1}^{n+m-1}\sum_{l=N+1,l\neq k}^{n+m-1}\sum_{i=1}^{n+m-1} V_{ki}\sum_{s=1}^{r}(V_{l,i+s} + V_{l,i-s}) = 0$$

$$(12)\ \tfrac{1}{1}((\sum_{k=1}^{n+m-1}\sum_{i=1}^{n+m-1} V_{ki}) - (m-1))^2 = 0$$

Za předpokladu, že platí $V_{ki} \in \{0,1\}$, rovnice 8 říká, že v každém řádku není více než jeden neuron s aktivací 1, jinými slovy $\sum_{i=1}^{n+m-1} V_{ki} <= 1$. Pro příklad, kdyby byl uzel k i-tý a j-tý navštívený: poté $V_{ki} = V_{kj} = 1$, tedy $V_{ki}V_{kj} = 1$ a součet by byl vyšší než 0. Pokud je však uzel k pouze i-tý navštívený, součet zůstane 0. Obdobně rovnice 9 říká, že v každém sloupci nesmí být více než jeden neuron s aktivací 1. Rovnice 10 je platná pouze v případě, že je počet aktivovaných neuronů roven $n + m - 1$, tedy počtu řádků a sloupců. Z kombinace tří předešlých podmínek vyplývá, že platné řešení musí obsahovat přesně jeden aktivovaný neuron v každém řádku a sloupci. [12]

Rovnice 10 se zabývá virtuálními depo uzly. Je platná pouze pokud nejsou virtuální depo uzly revidovány předtím než cesta mezi nimi obsahuje nejméně r dalších uzlů. Jinými slovy říká, že by cesty neměly být kratší než r. Bohužel tato podmínka nezaručuje minimální vzdálenost r i mezi virtuálním a reálným depem. Konečně rovnice 12 požaduje, aby posledních (m-1) řádků, reprezentujících virtuální depa, obsahovalo celkem (m-1) aktivních neuronů.

Metoda větví a mezí řeší celočíselný lineární program pomocí matice nul a jedniček. Hladový algoritmus a metoda simulovaného žíhání porovnává náhodné proveditelné řešení s podobnými náhodnými proveditelnými řešeními, dokud není dosaženo požadovaného konečného kritéria. Vzhledem k tomu, že neuronová síť modeluje elektrický okruh neuronů, jedná se zde o skutečné fyzikální zákony, které řídí změnu napětí v čase. Algoritmus neuronové sítě se spoléhá na fyzikální zákony pohybu, které Hopfield ustanovil pro v čase se měnící napětí na neuronech. Wacholder je dále modifikuje na časové deriváty pro napětí neuronů v MTSP neuronové síti, které používá k vytvoření pravidla iterativní aktualizace.

Nakonec je definována energetická funkce, která odpovídá rovnicím 8 až 12:

$$(13)\ E_1 = \tfrac{1}{2}\sum_{k=1}^{n+m-1}\sum_{i=1}^{n+m-1}\sum_{j=1 j\neq i} V_{ki}V_{kj}$$

$$(14) E_2 = \tfrac{1}{2}\sum_{i=1}^{n+m-1}\sum_{k=1}^{n+m-1}\sum_{l=1,l\neq k} V_{ki}V_{li}$$

$$(15) E_3 = \tfrac{1}{2}((\sum_{k=1}^{n+m-1}\sum_{i=1}^{n+m-1} V_{ki}) - (n+m-1))^2$$

$$(16) E_4 = \tfrac{1}{2}\sum_{k=n+1}^{n+m-1}\sum_{l=N+1,l\neq k}^{n+m-1}\sum_{i=1}^{n+m-1} V_{ki}\sum_{s=1}^{r}(V_{l,i+s} + V_{l,i-s})$$

$$(17) E_5 = \tfrac{1}{2}((\sum_{k=1}^{n+m-1}\sum_{i=1}^{n+m-1} V_{ki}) - (m-1))^2$$

s cílem minimalizovat:

$$(18) E_p = \tfrac{1}{2}\sum_{k=2}^{n}\sum_{l=1,l\neq k}^{n+m-1}\sum_{i=1}^{n+m-1} d_{kl}V_{ki}(V_{l,i+1} + V_{l,i-1})$$

s d, nákladovou maticí C, která zahrnuje (m-1) virtuálních depo bodů, pod podmínkou, že $E_1 = E_2 = E_3 = E_4 = E_5 = 0$. E_p byl mírně modifikován Wacholderovou definicí, takže se E_p rovná ceně bez cesty do a z depa. Početně, Wacholder minimalizuje funkci:

$$(19) J = E_p + \lambda_1 E_1 + \lambda_2 E_2 + \lambda_3 E_3 + \lambda_4 E_4 + \lambda_5 E_5,$$

kde λ_i je Lagranžův multiplikátor, který říká, jak je pravidlo i důležité. Například, pokud pravidlo 5 není příliš důležité, bude přiřazena malá hodnota λ_5; pokud pravidlo 5 není porušeno, pak $E_5 = 0$; ale pokud porušeno bylo, $E_5 >> 0$. Hodnota $\lambda_5 E_5$ má však stále malý vliv na celkový součet J. Pokud je pravidlo důležité, bude mu přiřazena vysoká hodnota λ_i. Přestože nastavení vysoké hodnoty λ_i může významně pomocí při hledání přípustného řešení, způsobí náklady E_p zcela irelevantní. Je tedy doporučeno zvolit počáteční hodnotu λ_i spíše nižší a v průběhu běhu algoritmu ji zvyšovat tak, aby bylo nalezeno přípustné řešení. [12]

V případě náhodné inicializace sítě je prvotní nastavení u_{00} určeno nastavením všech aktivačních rovnic.

$$u_{00} = g^{-1}(\tfrac{1}{n+m-1}) = u_0.tanh^{-1}(2.\tfrac{1}{n+m-1}),$$

kde u_0 je zvolený parametr. Některá forma symetrického odstraňování náhodného šumu se doporučuje pro zachování sítě v rovnovážném stavu. Každé U_{ki} je zpočátku nastaveno na $U_{ki} = u_{00} + \delta_{ki}$, s výrazem δ_{ki} označující šum náhodně vybraným z intervalu $[-0.1u_0, 0.1u_0]$.

Poté je vhodné nastavit, jak se v dalším kroku změní napětí U_{ki}. K tomu se používá pozměněné pravidlo pana Wacholdera. Používá funkci N úzce spojenou s parciální derivací J.

$$N(U_{ki} = -U_{ki}(\sum_{l=1,l\neq k}^{n+m-1} d_{kl}(V_{l,i+1} + V_{l,i-1})$$

$$+\lambda_1 \sum_{j=1,j\neq i}^{n+m-1} V_{kj} + \lambda_2 \sum_{l=1,l\neq k}^{n+m-1} V_{li}$$

$$+\lambda_3(\sum_{l=1}^{n+m-1}\sum_{j=1}^{n+m-1} V_{lj} - (n+m-1))$$

$$+\lambda_4(\sum_{l=n+1,l\neq k}^{n+m-1}\sum_{s=1}^{r}(V_{l,i+s} + V_{l,i-s}))$$

$$+\lambda_5(\sum_{l=n+1}^{n+m-1}\sum_{j=1}^{n+m-1} V_{lj} - (m-1))$$

pro $k = 1, .., n+m-1, i = 1, .., n+m-1$. Použitím Euler Forwardovi metody je možné každé U_{ki} a λ_i aktualizovat:

$$U_{ki}^{new} = U_{ki}^{old} + N(U_{ki}^{old}); \lambda_q^{new} = \lambda_q^{old} + E_q$$

pro $k = 1, .., n+m-1, i = 1, .., n+m-1, q = 1, .., 5$. Jak již bylo zmíněno dříve, všechny λ_q jsou po celý čas neklesající, takže se řešení bude s postupem času stále méně měnit. Algoritmus končí, když $\sum_{q=1}^{5} E_q$. [12]

Kapitola 4
Vlastní řešení

Všechna odběrná místa tvoří komplexní souvislý neorientovaný graf:

$G = (V, E, \epsilon)$,

kde V je neprázdná konečná množina odběrných míst (v řeči problému více obchodních cestujících měst), E je konečná množina cest mezi nimi a ϵ je přiřazení, které každé hraně $e \in E$ přiřazuje množinu $\{u, v\}$ (kde u,v $\in$ V jsou vrcholy) a nazývá se vztah incidence.

Existuje n odečítačů (tedy n obchodních cestujících).

Cílem řešení problému více obchodních cestujících je nalézt minimální kostru grafu.

Kostra grafu G je definována jako faktor grafu G (Podgraf $G' = (V', E', \epsilon')$ se nazývá faktor grafu G, jestliže $V' = V$.), který je stromem. Za předpokladu, že je dáno ohodnocení hran c, je minimální kostra grafu $G = (V, E, \epsilon)$ taková kostra grafu $K = (V, L, \epsilon)$, že $\sum_{e \in L} c(e)$ je nejmenší mezi všemi kostrami grafu G.

Pro nalezení minimální kostry grafu je použita kombinace algoritmů z kapitoly 3.

4.1 Aktuální stav

Ke grafickému zobrazení aktualního stavu byla použita poskytnutá data popsaná v Tabulce 1.2. Poskytnutá data v rámci odečtových jednotek, konkrétně kategorie Kód místa, GPS souřadnice X a GPS souřadnice Y. Z kódu místa je možné zjistit číslo výkonného pracovníka (2. a 3. číslice) a týden (4. a 5. číslice), ve kterém bylo dané místo odečteno. Konkrétně u příkladu, který je uveden v Tabulce 1.2. Poskytnutá data v rámci odečtových jednotek, 53740011-170102 je vidět, že toto místo odečetl výkonný pracovník číslo 37 ve 40. týdnu.

Společnost ČEZ Distribuce a.s. poskytla data severní části Olomouckého kraje. V této oblasti operují čtyři výkonní pracovníci s čísly 37, 38, 39, 40 a nachází se zde zhruba 110 000 odběrných míst. Všechna odběrná místa se nacházejí v oblasti zemského povrchu ohraničeného 49,59. a 50,45. rovnoběžkou a 16,74. a 18,17. poledníkem. Pro samotné grafické zobrazení v programovacím jazyce Java bylo použito rozšíření JPanel o velikosti 880 x 880 bodů. Navíc se počátek soustavy souřadné nachází v levém horním rohu a ne v levém spodním, jak je tomu ve výše zmíněné geografické oblasti. Bylo tedy nutné zavést následující přepočet geografických souřadnic na souřadnice použitelné v JPanelu:

$X_{JPanel} = 900 * (-16.6 + X_{geo})$
$Y_{JPanel} = 900 * (50.5 - Y_{geo})$

Odběrná místa odečtená jedním výkonným pracovníkem mají svoji specifickou barvu. Místa, která odečetl pracovník 37 jsou obarvena modře, místa odečtená pracovníkem 38 zeleně, pracovníkem 39 červeně, a konečně pracovníkem 40 oranžově. Dále je grafické zobrazení doplněno o 4 ukazatele mířící vždy na domovské místo výkonného pracovníka. Ve středu všech odběrných míst odečtených v jednom týdnu je zobrazeno číslo daného týdne. Pro lepší orientaci je na pozadí zobrazena mapa dané oblasti. Výsledné aktuální zobrazení je v Příloze B Aktuální stav.

4.2 Popis řešení

Výsledný program řešící zadanou úlohu, je implementovaný v programovacím jazyce Java. Při své funkci program ze zadaných vstupních dat (vstupní data mohou být omezená např. na určitý region) vypočítá optimální trasu pro všechny potřebné odečítače. Vstupními daty je myšlen seznam odečtových míst (včetně všech jejich parametrů, viz kapitola 1.2 Poskytnutá data), seznam odečítačů včetně jejích výchozího bodu a omezující podmínky vycházející z obchodní strategie a zkušeností společnosti ČEZ Distribuce a.s. Z těchto vstupních dat program vygeneruje výstup. Ten bude ve tvaru seznamu měst a doby, kterou má každý odečítač v daném městě strávit, pro všechny odečítače na každý pracovní den.

Program při své činnosti nejdříve načte všechny vstupy (seznam měst, seznam odečítačů, seznam uzlů silniční sítě a seznam silnic). Poté spočítá délku jednotlivých silnic ze zadaných údajů. Všechna načtená odběrná místa rozdělí dle doby od posledního odečtu na stará a nová místa. Nová místa jsou taková, která vznikla až po posledním odečtu dané oblasti a nebyla tak ještě nikdy odečtena. `Stará` místa (dále v textu jen místa) jsou všechna ostatní. Místa jsou dále rozdělena podle městské části. Tyto skupiny odběrných míst, kdy jedna skupina obsahuje všechna odběrná místa, která byla odečtena za jeden rok a patří pod konkrétní městskou část, se v dalším kroku dělí podle data odečtu a výkonného pracovníka. Nyní každá ze skupin obsahuje seznam míst, která byla odečtena v rámci jedné městské části jedním výkonným pracovníkem za jeden den.

Tato místa jsou dále seřazena dle času (hh:mm:ss) posledního odečtu, čímž vznikne posloupnost míst vzestupně seřazených dle času. Je-li časová mezera mezi dvěma místy větší než 1 hod (odečítač se např. zastavil na oběd), je tato posloupnost rozdělena na dvě nové, před pauzou a po pauze. Nyní se pomocí času odečtu prvního a posledního odběrného místa spočítá časová náročnost dané posloupnosti. Odfiltrují se posloupnosti, které obsahují méně než 10 odběrných míst. Sečtením časových náročností všech zbylých posloupností pro dané město se získá celková časová náročnost daného města (respektive městské části).

V této celkové časové náročnosti však ještě nejsou započítaná nová odběrná místa a posloupnosti obsahující méně než 10 odběrných míst. Pro každé z těchto míst si najdu nejbližší místo zahrnuté v nějaké posloupnosti (větší než 10 míst), spočítám si dobu, za kterou se k němu výkonný pracovník dostane (tam a zpátky), a spolu s dobou odečtu elektroměru o tuto hodnotu zvýším celkovou časovou náročnost dané městské části.

Nyní, když je známá celková časová náročnost odečtu každé městské části, je možné přistoupit k samotnému rozdělování odběrných míst mezi jednotlivé výkonné pracovníky. To probíhá pomocí kombinace dvou algoritmů a sice k-means a hladového algoritmu. Najde se nejbližší město (městská část) každého výkonného pracovníka, kde bude pracovat určitý počet hodin. Cílem je, aby každý výkonný pracovník pracoval denně 7.5 hod (0.5 je rezervováno na administrativu spojenou s odečty), čehož je docíleno tak, že se od 7.5 hod odečte dvojnásobek cesty z domovského místa výkonného pracovníka do

odečítaného města a zbytek času výkonný pracovník v tomto městě pracuje. Pokud by byla zbývající doba odečtu dané městské části (města) menší, než kolik zde má výkonný pracovník určeno na práci, přejede do města, které je nejblížší aktuálně odečítanému. Poté se do oněch 7.5 hod počítá cesta do prvního města, práce v prvním městě, přejezd do druhého města, práce v druhém městě a návrat domů z druhého města.

4.2.1 Předpoklady řešení

Toto řešení výkonným pracovníkům nediktuje přesnou posloupnost odběrných míst, které mají za den odečíst, ale pouze jim říká, ve kterém městě (městské části) mají daný den pracovat a jak dlouho. Je to tak z několika důvodů. Hlavním důvodem je předpoklad místní znalosti od daného výkonného pracovníka. Dá se předpokládat, že se člověk ve známém prostředí dokáže orientovat a pohybovat mnohem lépe a rychleji, než by to dokázal naplánovat počítačový algoritmus z omezeného množství parametrů. Tyto parametry se navíc mohou v čase měnit (například uzavírka silnice, kolona,..) a je tedy vhodnější, aby byl pohyb výkonného pracovníka v rámci lokální oblasti v jeho kompetenci.

Tato metoda určuje časovou náročnost jednotlivých městských částí převážně z jejich doby odečtu v roce 2017. Předpokládá tedy, že se doba odečtu v lokálním měřítku příliš nemění. Například, trval-li odečet ulice v roce 2017 pět hodin, dá se předpokládat, že se této hodnoty (se zanedbatelnou odchylkou v řádech procent) dosáhne i v letech následujících, neproběhnou-li zde zásadní změny. Nová místa (která nebyla v roce 2017 odečítána) se počítají samostatně, nebudou mít tedy na tuto skutečnost vliv.

4.2.2 Výhody a nevýhody řešení

Velkou výhodou tohoto řešení je, že není příliš radikální a výkonní pracovníci mají volnost při dělání dílčích změn. Pokud by řešení spočívalo v pevně naplánované posloupnosti měst, tak jak mají jít při odečtu za sebou a výkonný pracovník by v tomto řešení rád udělal nějaké drobné změny (např prohodil ulice, odečítal ulici z druhé strany, musel se vracet zpět k odběrnému místu, kde nebyl volně přístupný elektroměr a nezastihl zde majitele apod), muselo by se řešení (alespoň pro daný den) radikálně změnit, aby i nadále odpovídalo realitě. Navíc všechny velké zásahy do aktuálního stavu jsou z hlediska zavedení problematické a volnost řešení tyto změny alespoň trochu usnadňuje.

Odečet každého elektroměru trvá různě dlouhou dobu. Nedá se tedy říci, jak dlouho bude trvat odečet nějakého konkrétního elektroměru, jelikož tato doba závisí na řadě parametrů, které nelze všechny postihnout. Další výhodou této metody řešení je, že časové náročnosti odečtu jednotlivých městských části v sobě již zahrnují časy odečtu jednotlivých elektroměrů a není tedy nutné je odhadovat.

Na druhou stranu, výše zmíněné výhody přinášejí i několik nevýhod. Největší z nich je omezená míra možné optimalizace a tedy i prostoru pro zlepšení aktuálního řešení. Nově navrhnuté řešení je z části vázané na řešení akutální, kdy se tato část dá optimalizovat již jen ve velmi omezené míře. Navrhovanou metodu lze optimalizovat globálně, z pohledu časového rozvržení odečtu jednotlivých měst, ale téměř ji nejde optimalizovat lokálně, v rámci jednotlivých měst.

4.3 Mapy

Mapové podklady jsou důležitou součástí řešení pomocí výše zmíněné metody. Uplatní se při určování reálné vzdálenosti mezi výjezdním bodem výkonného pracovníka a odečítaným městem, či mezi dvěma a více odečítanými městy za jeden den. Dále se také uplatní při samotném plánování pořadí odečítaných měst.

4.3.1 Práce s mapami

Vhodný způsob reprezentace mapových podkladů pro tuto práci je například uzlový lokalizační systém. V tomto druhu reprezentace, podobnému grafovému znázornění, jsou dva druhy objektů- uzly a hrany. Uzly představují křížení cest a hrany jsou spojnice mezi uzly. Každý uzel je zadán unikátním identifikátorem (ID) a GPS souřadnicemi X a Y. Každá hrana je opět zadána unikátním identifikátorem (ID), ID počátečního a koncového uzlu a v některých případech i délkou. U každé hrany je tedy možné dle ID počátečního a koncového bodu dohledat i GPS souřadnice počátečního a koncového bodu. Existují dvě různé varianty reprezentace hran. V první variantě je hrana definována jako spojnice dvou uzlů, ze kterých vycházejí minimálně 3 cesty. U této varianty není možné spočítat délku cesty (jelikož se daná cesta může ve skutečnosti různě kroutit a je zadaný pouze její počáteční a koncový bod). Je tedy nutné, aby byla u této reprezentace zadaná i délka hrany. U druhé varianty je hrana definována jako spojnice uzlů, ze kterých vycházejí 2 a více cest. Takové hrany jsou rovné (úsečky) a lze u nich díky znalosti souřadnic počátečního a koncového bodu dopočítat délku dle Euklidovské vzdálenosti.

Velkou výhodou této reprezentace mapových podkladů je, že je velmi blízká běžně užívanému grafovému znázornění. Vzniklý stavový prostor je tedy možné prohledávat (hledat cestu mezi dvěma uzly) běžnými algoritmy, například A*. Nevýhodou naopak je, že je možné hledat cestu pouze mezi dvěma uzly. To znamená, že na krátké vzdálenosti, kdy je třeba určit vzdálenost bodů různých od uzlů, je tato metoda velmi nepřesná. Například pomocí této metody nelze určit vzdálenost dvou bodů na jedné hraně, jelikož cesta spojující tyto dva body neprochází žádným uzlem.

4.3.2 Získání vhodných mapových podkladů

Existují různé volně dostupné databáze mapových podkladů. Za zmínku stojí například Digitální vektorová geografická databáze České republiky ArcČR® 500 [1] v měřítku 1:500000, která vznikla ve spolupráci ARCDATA PRAHA, s.r.o., Zeměměřického úřadu a Českého statistického úřadu. Pro účely této práce je však tato databáze příliš málo podrobná a v nevhodném formátu. Server geofabrik.de [2] distribuuje volně dostupné geografické databáze založené na Open Street Map. Tyto databáze jsou již dostatečně podrobné, bohužel však pro účely vypracování této práce jsou opět v nevhodném formátu.

Přímo ze stránek www.openstreetmap.org [3] se dají exportovat geografické údaje zadané oblasti ve formátu XML. Jeden export je velikostně omezen na zhruba 50 MB. Velikost oblasti potřebné pro vypracování této práce je zhruba 1200 MB. Potřebná oblast tedy byla rozdělena na 24 dílčích oblastí, a pro každou zvlášť byl stažen soubor s geografickými údaji. Každý z 24 vygenerovaných souborů obsahuje geografická data ve formátu uzlového lokalizačního systému. Pokud nějaká hrana zasahuje do dvou a více dílčích oblastí, je obsažena v datech všech těchto oblastí. Výsledný soubor, který vznikl spojením všech 24 dílčích souborů, tedy obsahuje duplicitní uzly a hrany.

Pokud stavový prostor obsahuje duplicitní uzly, je jeho prohledávání velice ztíženo. Různé prohledávací algoritmy mohou daný konkrétní uzel považovat za již navštívený, ale uzel se stejnou polohou (a odlišným ID), do kterého míří duplicitní hrany, žádnou takovou značku nemá. Snadno se tedy může stát, že běh prohledávacího algoritmu skončí

[1] https://www.arcdata.cz/produkty/geograficka-data/arccr-500
[2] http://download.geofabrik.de/europe/czech-republic.html
[3] https://www.openstreetmap.org/export

kvůli tomuto problému neúspěchem. Při pokusu vymazat všechny duplicitní uzly (uzly s rozdílným ID, které mají stejnou polohu) se přeruší cesty vedoucí přes tento uzel.

4.4 Popis prvního návrhu

Program nejdříve načte všechny vstupy. Těmi jsou: odečtová místa, odečítači, mapové podklady a některé další dodatečné podmínky. Poté seřadí všechna odečtová místa dle doby posledního odečtu a to tak, aby místa s nejdelší dobou od posledního odečtu byla na začátku seznamu a nedávno odečítaná místa na konci. Dále se dle podobného principu každému místu přiřadí priorita odečtu a sice, že místo s nejdelší dobou od posledního odečtu bude mít největší prioritu a nedávno odečítaná místa nejmenší.

Z takto seřazených a ohodnocených míst se vezme část míst (např. všechna místa, kde nebyl prováděn odečet více než 11 měsíců) ze začátku seznamu (míst s nejdelší dobou od posledního odečtu a s nejvyšší prioritou), která se následně pomocí algoritmu K-means (kapitola 3.2) rozdělí do n skupin. Počet skupin n je možné získat vynásobením počtu pracovních dnů plánovaného časového rámce (např. měsíce) a počtu dostupných odečítačů, tedy: n = počet pracovních dnů * počet dostupných odečítačů. Počátační kapacita každé ze skupin se získá vydělením počtu vybraných míst a počtem skupin, tedy: kapacita = počet míst / n.

Nyní je program ve fázi, kdy je většina elektroměrů s dobou odečtu delší než zvolená konstanta, rozdělených do n skupin. Zbylé elektroměry, jejichž počet je menší, než je počet skupin, již nebyly rozděleny mezi skupiny z důvodu zachování stejného počtu elektroměrů v každé skupině. Pro každou skupinu program vyhledá nejkratší cestu pomocí genetického algoritmu. V ideálním případě by časová náročnost každé skupiny přesně pokryla pracovní dobu každého z odečítačů. To se však pravděpodobně nestane, vzniknou tedy tři druhy skupin: s delší než požadovanou časovou náročností, s požadovanou časovou náročností a s kratší než požadovanou časovou náročností.

Skupiny s nalezenou optimální trasou, která má délku požadované časové náročnosti, jsou již výstupem první části programu. Ve skupinách, kde je časová náročnost vyšší nebo nižší, než požadovaná, se vynásobí (a tedy změní) kapacita převrácenou hodnotou poměru reálné a požadované časové náročnosti. Tedy je-li například požadovaná časová náročnost 7 hodin a nějaká skupina má kapacitu c_{old} a časovou náročnost 10 hodin, změní se kapacita $c_{new} = c_{old} . \frac{7}{10}$. Snížením nebo zvýšením kapacity skupiny se změní počet míst ve skupině a tím pádem i časová náročnost dané skupiny. Nyní se znovu zjistí časové náročnosti nově vzniklých skupin a tento proces se opakuje do doby, než mají všechny skupiny požadovanou časovou náročnost.

V rámci počítačového programu se provedou tyto operace:

1 Nejdříve se sestupně seřadí všechna odběrná místa dle doby od posledního odečtu a každému místu se nastaví priorita odečtu (delší doba od posledního odečtu znamená vyšší prioritu).
2 Odběrná místa jsou shlukována do n shluků, kde n = počet pracovních dnů * počet dostupných odečítačů.
3 V každém shluku se hledá nejrychlejší trasa, která prochází všemi odečtovými místy daného shluku a vypočte se její časová náročnost dle genetického algoritmu.
4 U skupin s nižší nebo vyšší časovou náročností než požadovanou se vypočítá nová kapacita skupiny dle vztahu: $c_{new} = c_{old} . \frac{požadovanáčasovánáročnost}{reálnáčasovánáročnost}$.
5 Znovu se vypočte časová náročnost každé skupiny. Pokud jsou všechny v optimu, program končí. Pokud ne, pokračuje se bodem 3.

6 Výstupem programu jsou seřazené posloupnosti odběrných míst rozdělených do n skupin.

4.4.1 Problémy s prvním návrhem

Při implemetaci a tedy důkladnějším rozmyšlení výše uvedeného postupu se však narazilo na několik nedostatků. První problém je s filtrem dle doby posledního odečtu. Idea byla taková, že se seřadí všechna odběrná místa dle doby posledního odečtu a dle stejného principu se také všem místům přiřadí priorita. Ve výsledku tedy algoritmus primárně vybere místa s nejdelší dobou od posledního odečtu, ke kterým přiřadí geograficky nejbližší místa, byť s prioritou nižší. Avšak oblasti, které odečtou výkonní pracovníci za jeden týden v aktuálním rozdělení, spolu ve většině případů nesousedí. Nezřídka se stává, že jeden týden pracují výkonní pracovníci na jednom místě a příští týden pracují stovky kilometrů daleko. To má za následek vznik týdeních skupin odběrných míst, které se výrazně liší dobou od posledního odečtu a tedy i prioritou. Tím se radikálně snižuje možnost aktualní skupiny odběrných míst nějak významně přeorganizovat, jelikož když výkonný pracovník (nebo všichni 4 výkonní pracovníci, jak je to v aktuálním rozdělení) pojede do blízké oblasti s vysokou prioritou, je v podstatě vázán pouze na tuto oblast, protože okolní oblasti (které byly odečítany v jiném týdnu) mají výrazně nižší prioritu a tím i šanci, že budou přiřazeny k dané skupině.

Další problém byl s nastavováním kapacity. Původní idea byla, že se počáteční kapacita nastaví jako průměrný počet odběrných míst odečtených za jeden den (v tomto případě je to 120 odběrných míst za den), který se přenásobí koeficientem zohledňujícím požadovanou a skutečnou časovou náročnost dané skupiny, čímž se dostane ideální kapacita skupiny pro danou oblast. Aby nedošlo k situaci, kdy například bude výkonný pracovník odečítat obec se 125 odběrnými místy, ale dle požadované časové náročnosti vychází kapacita 122 odběrných míst, tedy zbyly by 3 odběrná místa, jejichž odečet nezabere příliš času, je požadovaná časová náročnost brána nikoliv jako přesná hodnota, ale jako časový interval (např. 6.5 - 7.5 hod). Problém této metody je však v nerovnoměrném osídlení. Metoda předpokládá jen málo se měnící hustotu osídlední, tedy že na jednotku ujeté vzdálenosti bude připadat podobný počet odběrných míst, což však v praxi může ale i nemusí platit. Dalším problém je samotný počet odběrných míst, které může za jeden den výkonný pracovník odečíst. V některých případech se může dostat až na přibližně 500 odběrných míst za den. Řešit problém obchodního cestujícího pro přibližně 500 míst hrubou silou je časově náročné. Avšak tato metoda předpokládá vysoký počet řešení TSP, jelikož se kapacita mění, dokud není časová náročnost ideální a s každou změnou kapacity je potřeba opět spočítat nejkratší trasu. Zbývá tedy použít algoritmy založené na náhodném, heuristickém postupu (v tomto případě genetický algoritmus), které však nevrací přesné řešení, ale nějaké náhodné dostatečně dobré. A jelikož se v každé iteraci mění skutečná časová náročnost, mění se i kapacita a proces se opakuje stále dokola.

4.5 Popis druhého návrhu

Jelikož bylo velmi obtížné sehnat vhodnou mapu České republiky, byla pro tvorbu této verze programu použita fiktivní mapa, připomínající mapu Kalifornie. Ta byla stažena z internetových stránek Utahské univerzity, konkrétně ze stránky: https://www.cs.utah.edu/ lifeifei/SpatialDataset.htm. Mapa je ve formátu uzlového lokalizačního systému. To znamená, že obsahuje seznam jednotlivých cest a uzlů s následující parametry:

Uzel: ID, zeměpisná délka, zeměpisná šířka
Hrana: ID, ID počátečního uzlu, ID koncového uzlu, délka hrany

U každého uzlu je tedy přesně zadaná jeho zeměpisná poloha a každá hrana má definováno, ve kterém uzlu začíná, ve kterém končí a jak je dlouhá. Dále bylo v blízkosti jednotlivých cest vygenerováno 10 000 míst a byla vygenerována maximální rychlost pro všechny cesty. Generovaná maximální rychlost byla 30, 50 nebo 90 km/hod.

Každé z 10 000 míst bylo přiřazeno k jedné konkrétní nejbližší silnici. Jednou z vlastností takto zadaných cest je, že každá cesta je reprezentována úsečkou z počátečního do koncového bodu. Obsahuje-li tedy například nějaká reálná cesta zatáčku, byla by v této reprezentaci rozdělena na několik kratších rovných úseků, které ve výsledku budou aproximovat danou zatáčku. Přiřazení bodů probíhalo pomocí vzdálenosti bodů od přímky, kdy byl bod přiřazen k nejbližší přímce (v tomto případě úsečce- cestě). Jelikož byly v tomto případě body vygenerovány buď přímo na přímku, nebo velmi blízko, nestalo se, že by byly přiřazeny k chybné cestě. Tento způsob rozdělení míst je inspirován skutečností, kdy domy (odběrná místa) vznikají v těsné blízkosti cest- ulic.

Dále byly určeny dva druhy ceny cesty. Prvním druhem cenou cesty se rozumí doba, za kterou je možné danou cestu projet, včetně zastávek u konkrétních míst, za účelem odečtu elektroměru. Pokud například bude cesta dlouhá 3km s maximální rychlostí 30km/hod a budou se na ní nacházet 3 odběrná místa, každé s dobou odečtu 6 min, bude přejezd po této cestě trvat: $\frac{3km}{30km/hod} + 3 * 0.1hod = 0.4hod$. Druhý druh ceny cesty předpokládá, že odběrná místa na této cestě byla již odečtena a výkonný pracovník tak touto cestou pouze projíždí bez zastavení za účelem odečtu. Na výše zmíněném příkladu by přejezd po takové cestě trval: $\frac{3km}{30km/hod} = 0.1hod$.

Nyní se původní problém přetransformoval v obdobu MTSP, pouze s tím rozdílem, že jednotliví obchodní cestující nemají za úkol navštívit všechna města, ale projít všechny cesty mezi nimi. Tento problém však není v této práci, z důvodu nedostatku času, dále řešen.

4.5.1 Problémy s druhým návrhem

Hlavním problémem tohoto způsobu řešení je už samotná reprezentace silniční sítě, která se zásadně liší od reálné silniční sítě. V reálné silniční síti je jako cesta uvažován úsek k tomu určený začínající jednou křižovatkou (křížením 2 a více cest) a končící jinou křižovatkou. Pokud se tedy výkonný pracovník dostane v reálném světě na konec cesty, má vždy minimálně dvě možnosti, jakou cestou dále pokračovat (za předpokladu, že jsou všechny cesty obousměrně průchozí). V této reprezentaci cesty je však reálná cesta rozdělena na několik úseků a nemusí tedy platit, že když se výkonný pracovník dostane na konec cesty (v této vygenerované reprezentaci), že bude mít na výběr minimálně ze dvou možností kam pokračovat. Ve většině případů bude mít jen jednu možnost a to pokračovat jedinou navazující cestou (nebo se vrátit). Při plánování na reálných cestách by mohl výkonný pracovník dorazit na konec cesty (křížení dvou a více cest), odkud může pokračovat jinou cestou, například zpět k domovu. V této reprezentaci však ve většině případů na konci konkrétní cesty může pouze pokračovat dále (k nějakému křížení), nebo se vrátit stejnou cestou.

Dalším probémem je, že u této metody řešení musí výkonný pracovník projít vždy celou cestu, aby odečetl odběrná místa na ní obsažená. To může být v některých případech nevhodné, například pokud by bylo odběrné místo na začátku dlouhé cesty, může být pro výkonného pracovníka výhodnější toto místo odečíst a vrátit se zpět, aniž by projel celou cestu.

Kapitola 5
Závěr

Tato práce se zabývá plánováním a optimalizací odečtových tras s využitím geografických informací. Nejprve bylo nutné nastudovat již existující algoritmy, které řeší dílčí problémy. Jejich přehled se nachází v kapitole 2 a 3. Kapitola 2 obsahuje úvod do problému obchodního cestujícího, jeho matematickou definici a na příkladu s počtem možných cest vzhledem k počtu měst názorně ilustruje, proč problém nelze řešit hrubou silou, ale je nutné hledat pro řešení problému sofistikovanější algoritmy.

Na začátku třetí kapitoly je popsán dvoufázový heuristický algoritmus řešící problém více obchodních cestujících. Jeho princip spočívá v rozdělení celého řešení na dvě části, přičemž v první části se množina všech měst alokuje do skupin, v rámci kterých při běhu druhé části řešení probíhá již samotné hledání nejkratší cesty. Jakožto zástupce algoritmů, které alokují jednotlivá města do skupin, je zde zmíněna vylepšená verze algoritmu K-means. Dále třetí kapitola obsahuje přehled algoritmů řešících samotný problém více obchodních cestujících.

Náplní čtvrté kapitoly je praktická část této práce. Nejdříve je zde popsáno zobrazení aktuálního stavu v grafické podobě. Ze zadaných dat v podobě GPS souřadnic jednotlivých odběrných míst a datem posledního odečtu jsou odběrná místa zobrazena na mapě, včetně barevného rozlišení jednotlivých odečítačů a popisem, v jakém týdnu byla ta která konkrétní odběrná místa odečtena. Vzniklý obrázek, zobrazující aktuální stav rozdělení, se nachází v příloze B.

Následuje popis vlastního řešení, včetně předpokladů, jeho výhod a nevýhod. Obdobně jako u aktuálního stavu, nachází se obrázek zobrazující navrhované rozdělení v příloze C. V navrhovaném rozdělení se podařilo odstranit několik problémů spojených s aktuálním řešením. Zásadním zlepšením je již vůbec existence softwaru, který dokáže globálně rozdělit odběrná místa mezi výkonné pracovníky vzhledem k jejich časovým možnostem a dalším parametrům popsaných v úvodu. Dalším bodem, který splňuje vytvořený program, je schopnost reagovat na fluktuaci zákazníků (tedy příbytek či úbytek odběrných míst) a na jejím základě upravit odečtové trasy jednotlivých výkonných pracovníků.

Další věcí, kterou se podařilo vylepšit oproti aktuálnímu řešení, je menší roztříštěnost obcí mezi jednotlivé odečítače a odečtové týdny. V navrhovaném řešení odečte výkonný pracovník celé město (městskou část) sám a to v nejkratší možné době. Teprve poté se přesune na jiné město, kde postupuje dle stejných pravidel. Toto opatření má za cíl eliminovat situace, kdy byla velká prodleva mezi odečty jednotlivých elektroměrů v rámci jedné obce. Nová odběrná místa jsou automaticky přiřazována do geograficky nejbližších odečtových skupin.

V implementaci řešení bohužel chybí mapové podklady. Všechny vzdálenosti jsou tedy počítány vzdušnou čarou. Tento nedostatek je kompenzován snížením průměrné rychlosti pohybu výkonných pracovníků. Při vytváření této práce bylo vyzkoušeno několik volně přístupných databází, včetně stažení mapových podkladů z Open Street Map. Detailněji je tento problém popsán v kapitole 4.3. Mapy. Mimo volně dostupné verze byl zaslán požadavek na Ředitelství silnic a dálnic ČR, zda by neposkytli mapu silniční

sítě Olomouckého kraje pro studijní účely a sice za účelem vypracování této práce. Bohužel zůstal tento mail, který je k nahlédnutí v příloze D, bez odpovědi. V implementaci jsou připravené jednotlivé metody, na práci s mapami. Změnit počítání vzdálenosti ze vzdušné na reálnou vzdálenost po silnici tak lze pouze nahráním mapových podkladů ve vhodném formátu a odkomentováním příslušných metod.

Literatura

[1] Jiri. Demel. *Grafy a jejich aplikace*. Vyd. 1 vydání. Praha: Academia, 2002. ISBN 978-802-0009-906.

[2] Keith J. Devlin. *Problémy pro třetí tisíciletí*. 1. vyd. v českém jazyce vydání. Praha: Dokořán, 2005. ISBN 80-720-3739-0.

[3] Xiaolong Xu, Hao Yuan, Mark Liptrott a Marcello Trovati. Two phase heuristic algorithm for the multiple-travelling salesman problem. *Soft Computing*. -. DOI 10.1007/s00500-017-2705-5.

[4] Jiří Šíma a Roman Neruda. *Teoretické otázky neuronových sítí*. Vyd. 1 vydání. Praha: MATFYZPRESS, 1996. ISBN 80-858-6318-9.

[5] *Biologické algoritmy (1) - Evoluční algoritmy*. `https://www.root.cz/clanky/biologicke-algoritmy-1-evolucni-algoritmy/`.

[6] Josef Hynek. *Genetické algoritmy a genetické programování*. 1. vyd vydání. Praha: Grada, 2008. ISBN 978-80-247-2695-3.

[7] Tolga Bektas. The multiple traveling salesman problem. *Omega*. 2006, 34 (3), 209-219. DOI 10.1016/j.omega.2004.10.004.

[8] C. E. Miller, A. W. Tucker a R. A. Zemlin. Integer Programming Formulation of Traveling Salesman Problems. *Journal of the ACM*. 7 (4), 326-329. DOI 10.1145/321043.321046.

[9] Yupo Chan a S.F. Baker. The multiple depot, multiple traveling salesmen facility-location problem. *Mathematical and Computer Modelling*. 2005, 41 (8-9), 1035-1053. DOI 10.1016/j.mcm.2003.08.011.

[10] Lawrence Davis. *Genetic algorithms and simulated annealing*. 1. publ vydání. London: Pitman u.a, 1987. ISBN 978-027-3087-717.

[11] Kevin Gurney. *An Introduction to Neural Networks*. London: Routledge, 1997. ISBN 978-020-3451-519.

[12] E. Wacholder, J. Han a R.C. Mann. A neural network algorithm for the multiple traveling salesmen problem. *Biological Cybernetics*. 1989, 61 (1), -. DOI 10.1007/BF00204755.

Příloha A
Použité zkratky, obsah přiloženého CD

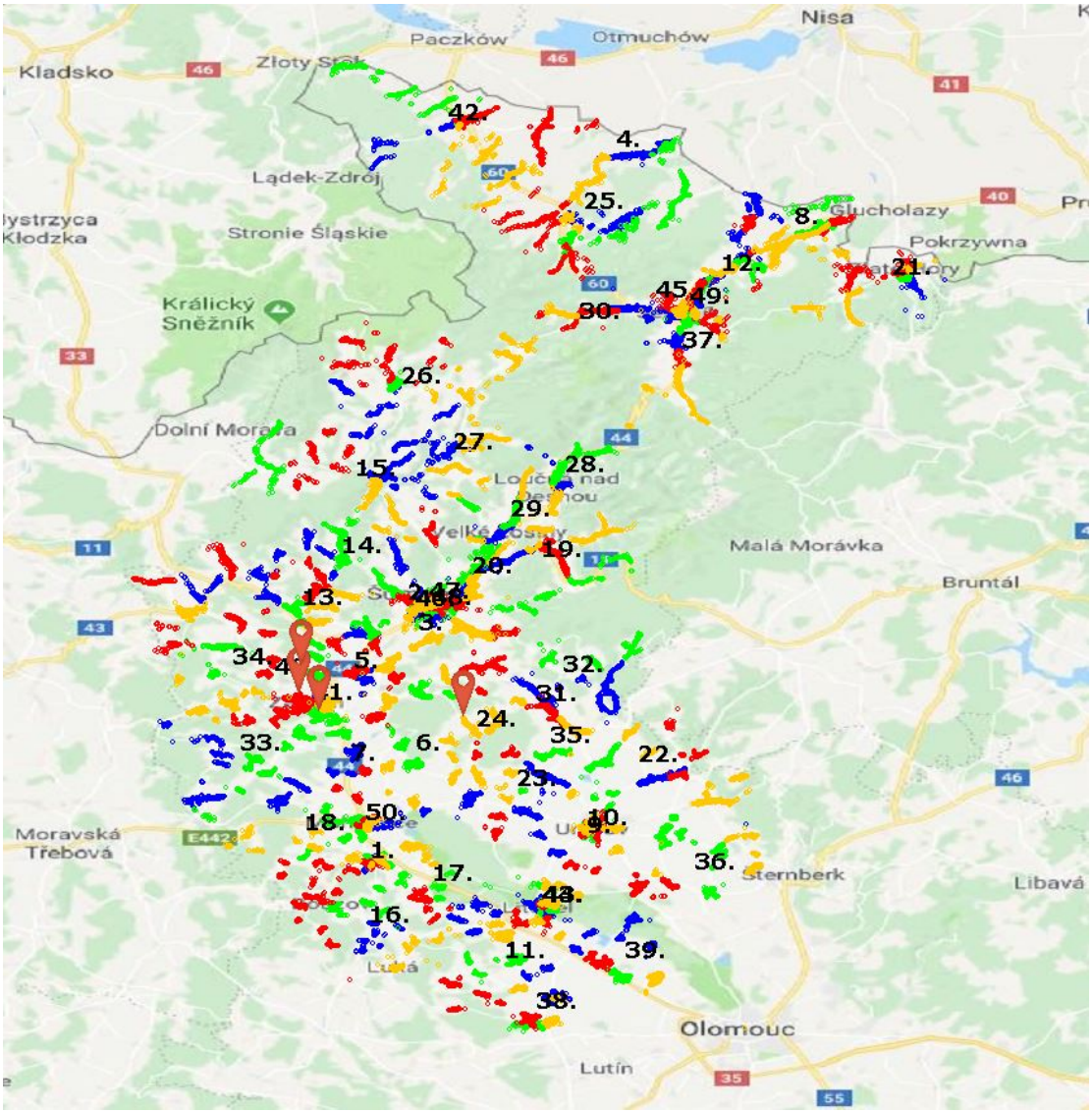
A.1 Zkratky

VČE	Východočeská energetika a.s.
STE	Středočeská energetická a.s.
SME	Severomoravská energetika a.s.
TSP	Traveling salesman problem
MTSP	Multiple traveling salesman problem
OJ	Odečtová jednotka
OM	Odběrné místo
TPHA	Two phase heuristic algorithm (dvoufázový heuristický algoritmus)
ILP	Integer linear program (celočíselný lineární program)

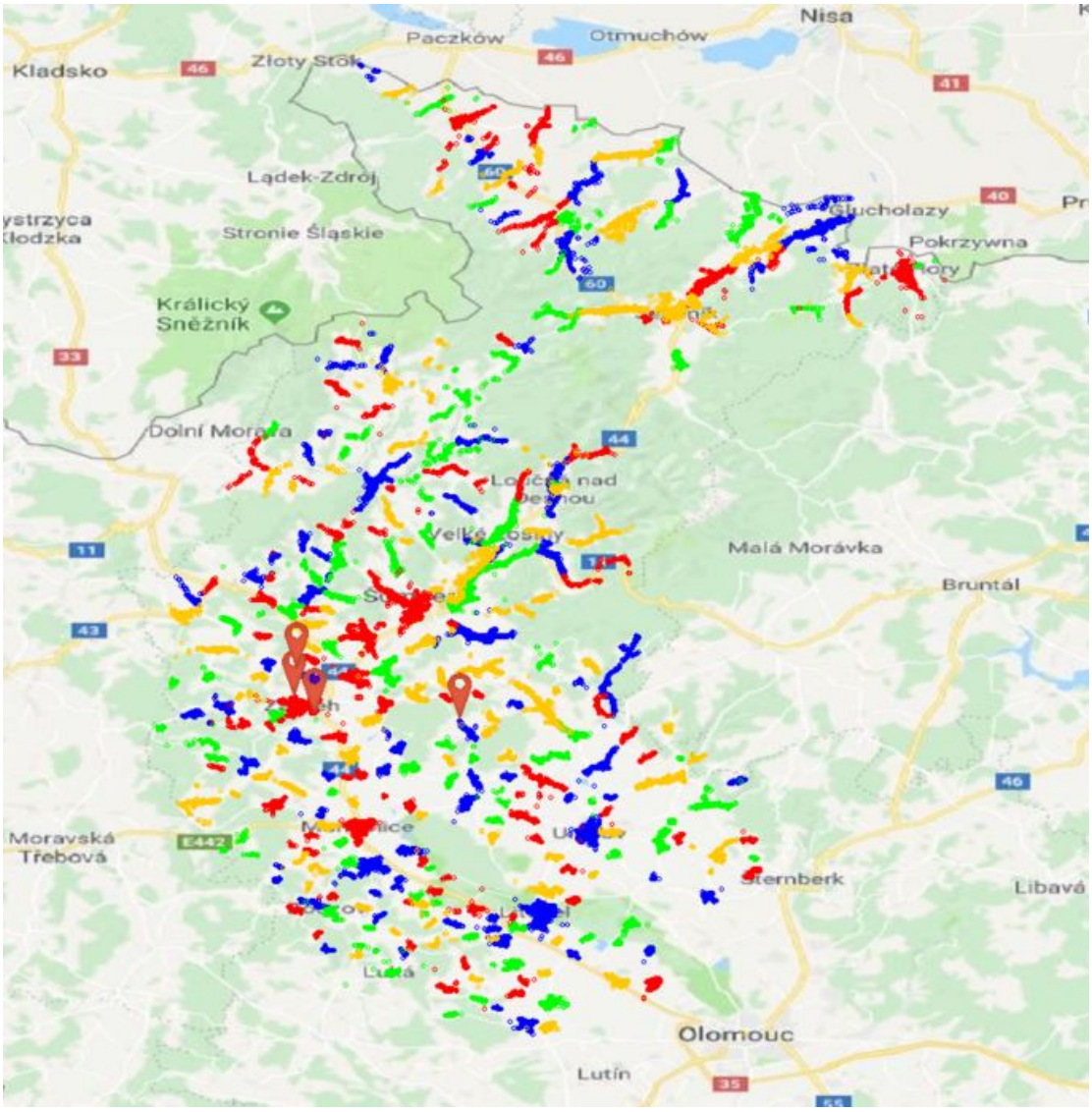
A.2 Obsah přiloženého CD

Zdrojové soubory implementovaného programu
BakalarskaPrace.pdf
mapa.jpg
edges.txt
nodes.txt

Příloha B
Aktuální stav



Příloha C
Navrhovaný stav



Příloha D

Žádost o poskytnutí mapových podkladů

24. 5. 2018 Seznam Email

stehudus@fel.cvut.cz 25. 4. 2018, 23:40

Komu: informace@rsd.cz

Datové podklady k bakalářské práci

Dobrý den,

jmenuji se Dušan Stěhule a jsem studentem třetího ročníku oboru Kybernetika a robotika na Fakultě elektrotechnické ČVUT. V tomto semestru se zabývám bakalářskou prací na téma "Plánování a optimalizace odečtových tras s využitím geografických informací", ve které zjednodušeně řečeno řeším problém více obchodních cestujících. To znamená, že plánuji nejkratší (/nejrychlejší) trasu pro odečítače takovou, aby navštívili všechna odběrná místa.

Jedním z bodů zadání je i experimentálně ověřit vyvinutý algoritmus na reálných datech a porovnat je se stávajícím řešením. Z jistých důvodů jsem si jako oblast pro porovnání stávajícího a mnou vyvinutého řešení vybral Olomoucký kraj. Bohužel se mi však nepodařilo najít žádné volně dostupné mapové podklady (v daném formátu), které jsou pro vypracování tohoto bodu nezbytné.

Chtěl bych Vás tedy požádat, zda by bylo možné poskytnout mi potřebná data. Konkrétně se jedná o Silniční a dálniční síť Olomouckého kraje ve formátu uzlového lokalizačního systému (ULS). Tato data by nebyla nikde zveřejněna, ani by nebyla použita ke komerčním účelům. Potřebuji je pouze pro studijní účely a sice dokončení bakalářské práce.

Předem děkuji za odpověď.

S pozdravem,
Dušan Stěhule

Přílohy

zadani_bakalarske_prace.pdf – PDF, 441 kB

Stáhnout Zobrazit ›