Vysoká škola ekonomická v Praze
Fakulta informatiky a statistiky
Katedra ekonometrie

Okružní a rozvozní úlohy

Habilitační práce

Autor: Ing. Jan Fábry, Ph.D.
Obor: Ekonometrie a operační výzkum

fabry@vse.cz

Praha, říjen 2014

Prohlášení k habilitační práci

Prohlašuji, že habilitační práci na téma „Okružní a rozvozní úlohy“ jsem vypracoval samostatně. Použitou literaturu a podkladové materiály uvádím v přiloženém seznamu literatury.

V Praze dne 1. října 2014

…….

Podpis

Poděkování

Rád bych touto cestou poděkoval prof. Ing. Josefu Jablonskému, CSc. za dlouholetou podporu a vytváření příznivých podmínek pro mou vědecko-výzkumnou činnost. Dále děkuji prof. RNDr. Janu Pelikánovi, CSc. za předané zkušenosti z oblasti diskrétní optimalizace a spolupráci při řešení praktických aplikací.

Věnování

Tuto práci věnuji památce mých rodičů.

Abstrakt

Okružní a rozvozní úlohy

Současné konkurenční prostředí vyvíjí přirozený tlak na firmy směrem k efektivním změnám a snižování nákladů. Předložená práce se týká především firem, které se zabývají přímo rozvozem či svozem osob, zboží, materiálu či odpadu, nebo zajišťováním služeb, při nichž je zapotřebí plánovat trasy vozidel. Danou problematikou se zabývají také firmy v rámci logistických procesů, které jsou nedílnou součástí jejich podnikání. Základním teoretickým východiskem v předloženém textu jsou úlohy obchodního cestujícího a listonoše včetně jejich modifikací. Vzhledem k šíři zkoumané problematiky je práce zaměřena na vybrané úlohy a jejich řešení pomocí exaktních matematických modelů i přibližných metod, které jsou v rozsáhlejších praktických aplikacích mnohdy jediným vhodným nástrojem pro nalezení efektivního řešení. Předmětem zájmu jsou především dynamické úlohy a problémy s více vozidly vyjíždějícími z jednoho či několika depotů.

Keywords: úloha obchodního cestujícího, rozvozní problem, úloha listonoše, úloha kurýrní služby, přepravní problém, dynamická úloha, trasy s více vozidly, heuristické algoritmy

Abstract

Vehicle and Arc Routing Problems

The current competitive environment creates natural pressure on companies toward effective changes and cost reduction. The work is mainly aimed at companies that are directly involved in persons, goods and material transporation and garbage collection, and firms providing service in which plans of vehicle routing are required. The problems are also issues for companies within the logistics processes that are an integral part of their business. The theoretical basis of the presented text is the travelling salesman problem and the postman problem, including their modifications. Because of wide range of problemacy, the work examines selected problems and their solving by exact mathematical models and approximate methods which are often the only suitable tool for finding appropriate effective solution to huge practical instances. Special emphasis is put on dynamic problems and problems with multiple vehicles located in one or multiple depots.

Keywords: travelling salesman problem, vehicle routing problem, arc routing probem, messenger problem, pickup and delivery problem, dynamic problem, multiple vehicle routing, heuristics.

Obsah

Úvod

Práce se zabývá oblastí logistických procesů týkající se generování tras především pro vozidla, která zajišťují obsluhu zákazníků či uspokojení jiných požadavků v nejrůznější podobě. Název práce vychází z české terminologie, o které pojednává následující kapitola. V obecné poloze lze považovat okružní úlohy a metody jejich řešení za východisko pro řešení rozvozních úloh. Proto je třetí kapitola věnována úloze obchodního cestujícího, na jejímž základě spočívají prakticky všechny další úlohy. Protože existuje nepřeberná řada modifikací této úlohy, ať již v deterministické či stochastické formě, není možné se v rámci jedné práce věnovat všem typům úloh, byť i v zjednodušené podobě. Považuji tudíž za významnější věnovat se podrobněji vybraným úlohám, kterými se zabývám ve své vědecko-výzkumné činnosti. To se týká především dalších kapitol a problémů, které víceméně na kapitolu o úloze obchodního cestujícího navazují.

Jak v úloze obchodního cestujícího, tak i v rozvozních úlohách je zvláštní důraz kladen na modifikace s několika vozidly vyjíždějícími z jednoho či několika výchozích míst. Speciální pozornost je věnována dynamickým úlohám, v nichž přicházejí nové požadavky po vyjetí vozidel na plánované trasy. Zařazování těchto požadavků vyžaduje velice rychlou reakci, a tudíž významnou roli v tomto případě hrají přibližné polynomiální metody. Jestliže je možné rozdělit dodávku do více tras, tedy dané místo obsloužit postupně několika vozidly, lze takovou úlohu řešit jako úlohu s dělenou dodávkou. I tato úloha, ostatně stejně jako ostatní zkoumané problémy, patří mezi NP-obtížné úlohy, a proto je opět nutné použít k řešení alternativní přístup v podobě heuristických algoritmů. Velice častou, a z hlediska praktického zcela přirozenou, modifikací okružních a rozvozních úloh je zavedení časových oken.

Další kapitola je věnována přepravním problémům, které jsou speciálními rozvozními úlohami. Poměrně triviální úlohou je Transhipment Problem, který se svojí podstatou řadí k tokovým úlohám na grafu. Největší část tvoří tzv. Pickup and Delivery problémy, mezi něž lze zařadit také úlohu kurýrní služby. Ta je studována v jednoduché podobě s jedním kurýrem, ale i v rozšířené verzi s několika vozidly vyjíždějícími jak ze stejného místa, tak i z několika různých míst. Heuristické postupy aplikované na tyto úlohy se stávají složitějšími, pokud je definována kapacita vozidel, případně jsou uvažována časová okna. V některých úlohách lze ovšem vystačit se zadáním časového limitu, který musí splnit každé vozidlo na trase. Také úlohu kurýrní služby lze definovat ve statické či dynamické verzi. Speciálním pře-

pravním problémem je úloha, v níž je nutné doručit většinou objemnější zásilky mezi omezeným počtem měst, kde každé z nich může být depotem. Cílem je naplánovat trasy a zároveň určit v těchto trasách výchozí místa. V takové úloze se většinou nelze vyhnout překládání naložených zásilek mezi vozidly. V práci je uveden originální model pro určení přepravních toků spolu s heuristickými postupy pro následné generování tras.

Zvláštní kapitola je věnována úloze listonoše a jejím modifikacím. Jedná se o tzv. Arc Routing Problem, který má mnoho praktických aplikací. V první části kapitoly je řešena známá úloha s jedním vozidlem bez jakéhokoli omezení. Cílem je nalézt Eulerův cyklus v grafu, a pokud takový cyklus neexistuje, pak zdvojit některé hrany grafu tak, aby následně nalezený Eulerův cyklus měl nejkratší délku. V případě, že je v úloze zavedeno kapacitní omezení, resp. časový limit, je nutné vygenerovat několik přípustných cyklů s cílem minimalizovat celkovou ujetou vzdálenost. Zajímavým případem je graf, v němž jsou hrany rozděleny na povinné a nepovinné. Zatímco povinnými hranami je nutné projet alespoň jednou, nepovinné hrany lze případně využít, pokud je to výhodné z hlediska celkové efektivity řešení. Protože také tyto úlohy patří do třídy NP-obtížných úloh, jsou předmětem mého výzkumu i v této oblasti heuristické algoritmy.

V poslední kapitole je prezentován průzkum týkající se softwarových produktů využívaných společnostmi či jinými subjekty pro plánování tras. Cílem kapitoly je demonstrovat, že v práci popsané modely a navržené metody jsou teoretickým východiskem pro reálné softwarové systémy, které jsou využívány pro každodenní tvorbu tras.

Jak již bylo uvedeno výše, cílem práce rozhodně není poskytnout dokonalý přehled o skutečně rozmanité oblasti okružních a rozvozních úloh, ale spíše prezentovat úlohy a jejich modifikace, které jsou v oblasti mého zájmu jak z hlediska vědeckého, tak z hlediska pedagogického. Konkrétním přínosem jsou modely a metody, které jsem publikoval jako autor či spoluautor v časopisech či ve sbornících mezinárodních konferencí. U většiny úloh je také naznačen směr, kterým se ubírá moje další vědecko-výzkumná činnost.

Hlavní přínos práce lze shrnout do následujících tří bodů:

1) Práce mapuje oblast okružních a rozvozních úloh, k nimž jsou formulovány matematické modely pro získání optimálních tras.

2) Protože se skutečně jedná o rozmanitou skupinu logistických problémů, jsou v práci vybrány konkrétní úlohy, které jsou předmětem mého vědeckého zájmu. Některé z úloh jsou

inspirovány praktickými problémy. Jedná se především o přepravní problém prezentovaný v podkapitole 4.3, pro jehož řešení byl formulován originální matematický model a navržen heuristický algoritmus pro generování tras, včetně určení depotů. Pro kapacitní problém listonoše, kterým se zabývá podkapitola 5.3, je navržen jak deterministický, tak stochastický heuristický postup, a to v několika verzích. Zvláštní pozornost je věnována úlohám kurýrní služby.

3) Vzhledem k NP-obtížnosti všech uvedených logistických problémů, jsou formulovány heuristické algoritmy pro řešení vybraných úloh v podobě modifikací známých heuristik (metody nejbližšího souseda, vkládacího algoritmu a metody výměn). Algoritmy jsou navrženy pro okružní a rozvozní úlohy s několika vozidly a úlohy kurýrní služby, a to jak jejich statické, tak i dynamické verze. V diskuzi je naznačen směr, kterým se ubírají úvahy o zvýšení efektivnosti navržených algoritmů, jež mohou vyústit v realizaci a vyhodnocení experimentů na reálných a vygenerovaných datech.

1 Vymezení oblasti okružních a rozvozních úloh

Orloff (1974) zavedl termín General Routing Problem, který v sobě zahrnuje Vehicle Routing Problem (VRP) a Arc Routing Problem (ARP). Některé publikace, obsahující všechny tyto problémy, často nesou alternativní název Network Routing (Ball et al., 1995). To je ovšem již poměrně široké označení, které kromě okružních a rozvozních úloh, může zahrnovat i optimalizační úlohy na grafech typu nejkratší cesta, nejdelší cesta, maximální zesílení, maximální profil, tokové úlohy, apod. V textu se tedy budeme držet klasického rozdělení úloh na VRP a ARP.

Pokud jde o vymezení VRP, většinou se pojem Vehicle Routing Problem překládá (a také definuje) jako „rozvozní problém“ či „rozvozní úloha“. Již samotné toto označení je nepřesné, neboť se jedná o úlohy, které samozřejmě zahrnují nejen rozvoz zboží, surovin, materiálu, lidí apod., ale také jejich svoz, nehledě na to, že mnoho praktických úloh dokonce připouští rozvoz a svoz v rámci jedné trasy (Pickup and Delivery Problem). Proto je nutné upozornit, že termín „rozvozní úlohy“ je zavádějící a bylo by vhodné jej nahradit přesnějším pojmem „úlohy rozvozu a svozu“. Bohužel ani tento termín neodpovídá anglickému výrazu „routing problems“, neboť vytváření tras (routing[1]) nemusí být nutně spojeno s „rozvozem“ či „svozem“, ale pouze s „průjezdem“ danými místy či úseky, resp. s jejich „návštěvou“. K praktickým úlohám tohoto typu se řadí například opravy či revize, odečty elektroměrů a plynoměrů aj. V takovém případě nelze hovořit o rozvozních úlohách či úlohách svozu, a proto jsou v práci označovány jako „okružní úlohy“, k nimž se řadí úloha obchodního cestujícího (Travelling Salesman Problem, TSP) spolu se všemi jejími modifikacemi.

Bohužel ani anglický termín Vehicle Routing Problem není zcela přesný, neboť trasa nemusí být nutně realizována vozidlem. Zřejmě i z tohoto důvodu někteří autoři (např. Ball et al., 1995) raději používají výše uvedené označení Network Routing, které je spojeno s vytvářením tras na grafu. Lze se pak setkat i s pojmy Node Routing a Arc Routing. První pojem, který má velice blízko k úlohám TSP a VRP, se týká vytváření okruhů přes uzly, druhý pojem je spojen s generováním tras přes hrany. Nejznámější úlohou tohoto typu je úloha (čínského) listonoše, která má mnoho praktických aplikací a z nich vyplývajících modifikací optimalizačních modelů či speciálně vytvořených algoritmů pro jejich řešení. Pro termín Arc

[1] Vzhledem k tomu, že termín „routing“ lze přeložit jako „trasování“, můžeme zřejmě za nejvýstižnější český termín pro úlohy, které jsou předmětem této práce, považovat pojem „trasovací úlohy“ (Janáček, 2003), a to i přesto, že se jedná o označení obecnějšího charateru.

Routing nenajdeme v české terminologii vhodný ekvivalent, konkrétní úlohy lze ovšem zařadit mezi okružní úlohy, případně úlohy rozvozu a svozu.

Existuje mnoho variací úlohy TSP (Punnen, 2002). Předně, úlohu lze formulovat jako symetrickou nebo nesymetrickou (viz Kapitola 2). V některých modifikacích lze předpokládat závislost nákladů spojených s přejezdem či dob přejezdu mezi místy na čase (Time Dependent TSP). V periodickém TSP (Solomon a Desrosiers, 1988) se jedná o plánování tras v rámci předem určeného časového horizontu, např. několika dní, během nichž je pro každé místo určen počet nutných návštěv. Cílem je naplánovat denní trasy tak, aby byla minimalizována délka všech tras. V další variaci „Black and White TSP" (Ghiani et al., 2006) jsou uzly rozděleny na „černé" a „bílé" a je nutné respektovat dvě pravidla: 1) počet bílých uzlů v trase mezi jakýmikoliv dvěma po sobě následujícími černými uzly nesmí překročit předem dané číslo, 2) vzdálenost mezi jakýmikoliv dvěma po sobě následujícími černými uzly musí respektovat maximální povolenou hranici. V selektivním TSP (Gendreau et al., 1998b) je každému uzlu přiřazena váha (preference). Cílem je navrhnout trasu přes vybrané uzly tak, aby součet jejich vah byl maximální při respektování povoleného limitu pro délku trasy. Zajímavou modifikací je úloha, v níž je zadána množina dvojic míst, z nichž první místo musí být navštíveno před místem druhým. Speciálním případem je úloha kurýrní služby (viz podkapitola 4.2).

V praktických úlohách se velmi často objevuje požadavek na respektování časových oken jednotlivých zákazníků, což jsou intervaly, v nichž je lze navštívit. Další rozsáhlý okruh tvoří problémy s více obchodními cestujícími, kteří obsluhují zákazníky z jednoho či několika výchozích míst (viz podkapitola 2.5). Své výpočetní zkušenosti shrnuli Svestka a Huckfeldt (1973). V počátcích výzkumu se zapsal výrazně Gavish (1976), významná je především pozdější práce autorů Gavish a Srikanth (1986), kteří navrhli metodu pro nalezení optimálního řešení pro výše zmíněné úlohy větších rozměrů. Podobně Laporte a Nobert (1980) navrhli pro řešení této úlohy algoritmus založený na metodě řezných nadrovin. Systém DRIVE[2] založený na metodě větvení a oceňování[3] popsali Savelsbergh a Sol (1998).

Pokud jsou všichni zákazníci známí před vlastní optimalizací, resp. před zahájením realizace tras, jedná se o tzv. statickou úlohu TSP. V opačném případě, tj. pokud připouštíme zařazení nových požadavků do naplánovaných tras po vyjetí vozidla či vozidel, definujeme tzv. dynamickou úlohu TSP. Z hlediska pozdějšího zkoumání jsou významné především následu-

[2] DRIVE je zkratkou názvu Dynamic Routing of Independent VEhicles.
[3] Metoda větvení a oceňování (Branch-and-Price Algorithm) využívá metodu generování sloupců.

jící práce: Psaraftis (1988, 1995), Powell et al. (1995) a Lund et al. (1996). Larsen et al. (2004) provedli řadu zajímavých simulačních experimentů pro úlohu obchodního cestujícího s apriorní informací v podobě pravděpodobnosti vzniku požadavku v určitém regionu.

V rozvozních úlohách je významná kapacita vozidla vzhledem k nenulovým požadavkům obsluhovaných míst. V typickém případě je svoz či rozvoz rozvržen do několika tras, které mohou být realizovány paralelně (pokud je k dispozici více vozidel) či sériově (postupně jedním vozidlem). Jednou z nejvýznamnějších publikací, obsahující přehled základních variant rozvozních úloh včetně metod pro jejich řešení, je sborník Totha a Viga (2002). Stejně jako v úloze obchodního cestujícího, také v rozvozní úloze mohou být zadána časová okna (Savelsbergh, 1992). Ve výše uvedeném sborníku Cordeau et al., 2002 popisují získání horních odhadů účelové funkce pomocí speciálních heuristických algoritmů za pomoci dekompozičních přístupů. Kromě pevně daných časových oken (hard time windows) definují i možnost jejich uvolnění (soft time windows). Solomon a Desrosiers (1988) formulovali rozvozní úlohu, v níž každý zákazník musí být navštíven několikrát během určitého časového horizontu (multiple time windows).

Další významnou publikací je sborník Goldena et al. (2008), který obsahuje 25 prací týkajících se pokroku v oblasti rozvozních úloh včetně prezentace budoucího výzkumu. Jedná se především o úlohy s rozdílnou kapacitou vozidel, dynamické rozvozní úlohy a různé varianty přepravního problému (Pickup and Delivery Problem); speciálními úlohami tohoto typu jsou Dial-a-Ride Problem (Cordeau a Laporte, 2003) a úloha kurýrní služby (viz podkapitola 4.2). Ve sborníku je prezentována zajímavá úloha, která propojuje rozvozní úlohy s modely řízení zásob (tzv. Inventory Routing, Bertazzi et al., 2008). Další variantou je rozvozní úloha se stochastickou poptávkou, kterou rozpracovali Gendreau et al. (1996). Stejní autoři také navrhli další rozšíření metod na situace, kdy lze např. využít dostupné informace o budoucích požadavcích. Na dalším vývoji metod řešících dynamické rozvozní úlohy se podíleli Gendreau a Potvin (1998), kteří předložili přehled základních aplikací z této oblasti. Gendreau et al. (1999) zkoumali dynamickou rozvozní úlohu s časovými okny na příkladu kurýrních služeb, zajišťujících požadavky zákazníků, které je nutné navštívit během časového okna, vyzvednout připravenou zásilku a dovézt na požadované místo, případně do centra, kde se zásilky shromažďují pro další zpracování, případně hromadný rozvoz (např. služba DHL).

Pro řešení rozsáhlejších úloh, časově velmi náročných, lze použít řadu heuristických postupů, které poskytují poměrně kvalitní řešení v akceptovatelném čase. V práci Gendreau et

al. (1999) byla použita paralelní metoda tabu search. Pro statickou úlohu popsali podobný heuristický algoritmus Badeau et al. (1997) a Taillard et al. (1997). Montemanni et al. [127] použili pro dynamickou rozvozní úlohu metaheuristický algoritmus[4] založený na chování mravenčí kolonie (tzv. Ant Colony System). Gambardella et al. [126] představili dva softwarové produkty vycházející ze zmíněného algoritmu. Použitím genetického algoritmu (Genetic Algortihm) a metody založené na principu simulovaného žíhání (Simulated Annealing) v rozvozní úloze s časovými okny se zabývají Thangiah et al. [128] a Bräysy a Gendreau (2005b). Aplikaci genetického algoritmu na tento typ úlohy popisují také Homberger a Gehring (1999), Potvin a Bengio (1996), Čičková (2005) aj.

V dnešní době jsou předmětem zájmu alternativní optimalizační postupy pro řešení časově náročných úloh, založené na metodě větvění a řezů či generování sloupců. Metoda větvení a řezů[5] pro statickou rozvozní úlohu je popsána v práci Bard et al. (2002) a Achuthan et al. (2003). Pro statickou rozvozní úlohu s časovými okny použili metodu generování sloupců[6] Desrochers et al. (1992), pro dynamickou úlohu Chen a Xu (2006).

V reálných rozvozních úlohách velmi často existuje možnost rozdělení dodávky zákazníkovi mezi několik vozidel, resp. mezi více tras. Touto problematikou se zabývají Archetti et al. [125], Dror a Trudeau (1989), Dror et al. (1994), Fábry (2005a) aj. Metodu tabu search aplikovali na tuto úlohu Archetti et al. (2006). Souhrnný přehled exaktních metod, klasických heuristických algoritmů a metaheuristických metod pro řešení rozvozních úloh nabízí již zmíněný sborník Totha a Viga (2002).

Poslední oblastí, kterou se zabývá tato práce, jsou Arc Routing Problems, mezi něž se řadí úloha listonoše v nejrůznějších modifikacích. K významným pracím patří především dvě publikace Eiselta et al. (1995a, 1995b) a sborník Drora (2000). Najdeme zde přehled všech nejznámnějších variant zmíněné úlohy včetně realných aplikací, matematických modelů a heuristických postupů pro jejich řešení. Pro kapacitní úlohu listonoše, která je předmětem této práce, vyvinuli Hertz et al. (2000) metaheruristický algoritmus typu Tabu Search, Santos et al. (2010) prezentovali metaheuristiku založenou na mravenčích koloniích (Ant Colony). Longo et al. (2006) řeší tuto úlohu její transformací na VRP.

[4] Metaheuristické algoritmy jsou obecné heuristické postupy aplikovatelné na obecný optimalizační problém, zatímco algoritmy označované jako heuristické jsou speciální metody formulované pro konkrétní úlohu.
[5] Branch-and-Cut Algorithm.
[6] Column Generation Algorithm.

2 Úloha obchodního cestujícího a její modifikace

V základním problému obchodního cestujícího (TSP)[7] je cílem najít okruh, v němž je každé místo (včetně místa výchozího) navštíveno právě jednou a celková vzdálenost, kterou vozidlo ujede, je minimální. Řešení této optimalizační úlohy předpokládá apriorní znalost všech míst[8] (např. zákazníků) a nejkratších vzdáleností mezi každou dvojicí míst. Tato základní verze TSP má řadu modifikací a rozšíření (Gutin a Punnen, 2002).

2.1 Nesymetrická úloha obchodního cestujícího

Nechť je zadáno *n* míst, včetně výchozího místa, které je označeno číslem 1. Pro každou dvojici míst *i* a *j* je definována nejkratší vzdálenost c_{ij} $(i, j = 1,2,...,n)$. Pokud předpokládáme, že matice nejkratších vzdáleností nemusí být symetrická, je nutné k takto zadanému problému TSP přistoupit jako k tzv. *nesymetrické úloze obchodního cestujícího* (Pelikán, 2001). Matematický model úlohy lze formulovat takto:

$$\text{minimalizovat} \quad z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}, \tag{2.1}$$

za podmínek

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1,2,...,n, \tag{2.2}$$

$$\sum_{i=1}^{n} x_{ij} = 1, \quad j = 1,2,...,n, \tag{2.3}$$

$$u_i - u_j + nx_{ij} \le n-1, \quad i = 1,2,...,n, \quad j = 2,3,...,n, \tag{2.4}$$

$$x_{ij} \in \{0,1\}, \quad i, j = 1,2,...,n, \tag{2.5}$$

$$u_i \in R_0^+, \quad i = 1,2,...,n. \tag{2.6}$$

Binární proměnná x_{ij} nabývá hodnoty 1 v případě, že vozidlo navštíví místo *j* bezprostředně po návštěvě místa *i*, hodnoty 0 v opačném případě. Účelová funkce (2.1) představuje celkovou vzdálenost, kterou vozidlo ujede. Soustavy rovnic (2.2) a (2.3) zajišťují, že každé místo je v rámci okruhu navštíveno právě jednou. Nerovnice (2.4) s nezápornými proměnnými u_i, označované jako smyčkové podmínky, zabrání vzniku parciálních cyklů (Miller et al., 1960).

[7] Travelling Salesman Problem.
[8] V některých případech tzv. dynamické úlohy nejsou všechna místa známa předem - viz podkapitola 2.6.

Znázorníme-li úlohu pomocí úplného grafu, pak jejím optimálním řešením je Hamiltonův cyklus s minimálním součtem ohodnocení vybraných hran.

V celé práci budeme z důvodu zjednodušení všech matematických modelů předpokládat, že proměnné x_{ij} jsou definované i pro $i = j$. V opačném případě bychom ve všech účelových funkcích, omezeních a matematických výrazech museli připojit podmínku pro indexy $i \neq j$. Tento postup by byl jistě korektní, ovšem na úkor přehlednosti. Nicméně i v tomto případě lze snížit počet nerovnic (2.4) o hodnotu (n – 1), pokud v nich použijeme výše zmíněnou podmínku $i \neq j$. V takovém případě je ovšem nutné, aby matice nejkratších vzdáleností neobsahovala na diagonále nuly, ale dostatečně vysokou prohibitivní konstantu, v celé práci označovanou *M*. Desrochers a Laporte (1991) ukázali, že smyčkové podmínky (2.4) lze zesílit jejich následující úpravou:

$$u_i - u_j + (n-1)x_{ij} + (n-3)x_{ji} \leq n-2, \quad i = 1,2,...,n, \quad j = 2,3,...,n. \tag{2.7}$$

2.2 Symetrická úloha obchodního cestujícího

V předchozí části jsme předpokládali, že matice nejkratších vzdáleností mezi místy nemusí být symetrická. Jestliže platí $c_{ij} = c_{ji}$ $(i, j = 1,2,...,n)$, je možné TSP řešit jako tzv. *symetrickou úlohu obchodního cestujícího* (Pelikán, 2001). Vzhledem k tomu, že matice nejkratších vzdáleností je symetrická, lze definovat proměnné x_{ij} jen pro $i < j$, tedy pro $i = 1,2,...,n-1,\ j = i+1, i+2,...,n$. Hledáme-li Hamiltonův cyklus v úplném neorientovaném grafu, pak $x_{ij} = 1$, jestliže hrana (i, j) je zahrnuta v cyklu. V opačném případě je $x_{ij} = 0$. Účelovou funkci (2.1) lze pak zapsat v následujícím tvaru:

$$\text{minimalizovat} \quad z = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} x_{ij}. \tag{2.8}$$

Podmínky (2.2) a (2.3), které zaručují, že každé místo bude vozidlem navštíveno právě jednou, vyjádříme touto soustavou rovnic:

$$\sum_{j=1}^{i-1} x_{ji} + \sum_{j=i+1}^{n} x_{ij} = 2, \quad i = 1,2,...,n. \tag{2.9}$$

Eiselt a Sandblom (2000) uvádějí dvě základní možnosti definice podmínek[9], zabraňujících vzniku parciálních cyklů, vycházející z formulace úlohy Dantzigem et al. (1954).

První možnost je založena na faktu, že v cyklu se počet hran rovná počtu uzlů. Označme množinu všech míst (uzlů) $U = \{1,2,...,n\}$. Podmínky lze pak formulovat následujícím způsobem:

$$\sum_{\substack{i \in U' \\ i<j}} \sum_{j \in U'} x_{ij} \leq |U'| - 1, \quad \forall U' \subset U, \quad 3 \leq |U'| \leq \left\lfloor \frac{n}{2} \right\rfloor, \tag{2.10}$$

kde $|U'|$ je počet uzlů v množině U' a $\left\lfloor \frac{n}{2} \right\rfloor$ je celá část čísla $\frac{n}{2}$. Kdyby existoval parciální cyklus, obsahující všechny uzly z podmnožiny U', tvořilo by jej $|U'|$ hran. Aby bylo toto nebezpečí eliminováno, smí pro danou podmnožinu uzlů existovat nejvýše $|U'|-1$ hran spojujících tyto uzly. Výše uvedené podmínky platí pro všechny podmnožiny $U' \subset U$, v nichž by mohl parciální cyklus teoreticky vzniknout. Vzhledem k definici binárních proměnných x_{ij} $(i < j)$, není nutné v podmínkách uvažovat podmnožiny s jedním či dvěma uzly.

Druhá možnost, jak zabránit vzniku parciálních cyklů, vychází ze souvislosti výsledného grafu, tedy Hamiltonova cyklu. Pokud by výsledkem byly parciální cykly, pak nutně neexistují žádné hrany, které by je spojovaly. Podmínky (2.10) lze tedy alternativně zapsat jako

$$\sum_{\substack{i \in U' \\ i<j}} \sum_{j \in U \setminus U'} x_{ij} + \sum_{\substack{i \in U \setminus U' \\ i<j}} \sum_{j \in U} x_{ij} \geq 1, \quad \forall U' \subset U, \quad 3 \leq |U'| \leq \left\lfloor \frac{n}{2} \right\rfloor \tag{2.11}$$

Pro všechny dvojice množin uzlů U' a $U \setminus U'$ (doplněk množiny U' v množině U), v nichž by mohl teoreticky vzniknout parciální cyklus, musí existovat alespoň jedna hrana, která tyto dvě množiny uzlů spojuje, tj. alespoň jedna z proměnných x_{ij}, kde i je index uzlu z jedné množiny a j index uzlu z množiny druhé, nabývá hodnoty 1. Výraz na levé straně nerovností (2.11) musí mít uvedenou podobu opět vzhledem k definici proměnných x_{ij} $(i < j)$.

Modifikace úlohy obchodního cestujícího uvedené v dalším textu jsou založeny na nesymetrické úloze.

[9] V uvedené publikaci jsou podmínky formulovány s plnou maticí proměnných x_{ij}, nikoli s proměnnými definovanými pouze pro $i < j$.

2.3 Úloha obchodního cestujícího s časovými okny

Velmi častým rozšířením standardní úlohy TSP je přidání časových oken pro některá nebo všechna místa[10]. Jedná se o časový interval $\langle e_i, l_i \rangle$, v němž je nutné místo *i* navštívit. Dolní mez intervalu e_i je tedy nejdříve možným termínem příjezdu vozidla do místa *i*, horní mez intervalu l_i je pak nejpozději přípustným termínem příjezdu do tohoto místa. Kromě nejkratší vzdálenosti mezi dvojici míst *i* a *j* je nutné znát také dobu přejezdu z místa *i* do místa *j*, kterou označíme t_{ij}. Zavedeme-li proměnnou τ_i, jejíž hodnota odpovídá okamžiku příjezdu vozidla do místa *i*, pak nutně musí platit $\tau_i \in \langle e_i, l_i \rangle$. Protože v úloze obchodního cestujícího s časovými okny (TSPTW)[11] hraje důležitou roli čas, je nutné pro každé místo navíc definovat dobu S_i, kterou vozidlo, resp. jeho posádka, stráví v místě *i*.

Matematický model takto rozšířené úlohy je založený na modelu (2.1) – (2.6), v němž jsou smyčkové podmínky (2.4) nahrazeny následujícími nerovnostmi:

$$\tau_i + S_i + t_{ij} - M(1 - x_{ij}) \leq \tau_j, \quad i = 1,2,...,n, \quad j = 2,3,...,n, \tag{2.12}$$

kde symbol *M* představuje již dříve zmíněnou vysokou konstantu. Dále jsou přidány omezující podmínky respektující časová okna a je stanoven okamžik výjezdu vozidla z výchozího místa. Proměnné u_i jsou tedy v modelu nahrazeny nezápornými proměnnými τ_i:

$$e_i \leq \tau_i, \quad i = 2,3,...,n, \tag{2.13}$$

$$\tau_i \leq l_i, \quad i = 2,3,...,n, \tag{2.14}$$

$$\tau_1 = 0. \tag{2.15}$$

Desrosiers et al. (1995) uvádějí podmínky (2.12) také v následující podobě:

$$x_{ij}(\tau_i + t_{ij} - \tau_j) \leq 0, \quad i = 1,2,...,n, \quad j = 2,3,...,n. \tag{2.16}$$

Je zřejmé, že tato omezení jsou nelineární a pro praktické využití tudíž vyhovují lépe původní omezení.

Podmínky (2.13) a (2.14), které jsou v literatuře označované jako „hard“, striktně vyžadují dodržení všech časových oken. V některých případech lze uvolnit podmínky (2.14), tj. vozidlo smí přijet do daného místa až po „uzavření“ časového okna, ovšem za cenu stanoveného penále, které může být funkcí velikosti zpoždění vozidla. Takto uvolněné podmínky se větši-

[10] Dále budeme předpokládat, že časová okna jsou definována pro všechna místa kromě místa výchozího.
[11] Travelling Salesman Problem with Time Windows.

nou nazývají „soft“ (Gendreau et al., 1999). Časová okna zákazníků mohou být v reálné situaci definována tak, že během intervalu, kdy je okno „otevřené“, je nutné k zákazníkovi nejen přijet, ale také dokončit jeho obsluhu. V takovém případě, stačí upravit podmínky (2.14) následujícím způsobem:

$$\tau_i + S_i \leq l_i \quad i = 2,3,...,n. \tag{2.17}$$

Zadanou hodnotu l_i lze pak interpretovat jako nejpozději přípustný konec obsluhy zákazníka i.

V souvislosti s časovými okny, resp. časem jako významným faktorem při vytváření trasy, je nutné přemýšlet o tom, jak se zachová posádka vozidla v případě, že doba přejezdu mezi dvěma po sobě navštívenými místy je kratší než délka intervalu mezi termínem nejdříve možného příjezdu do místa a okamžikem, kdy vozidlo odjede z předchozího místa. Mohou nastat dvě základní možností: vozidlo počká u právě obslouženého zákazníka nebo dojede co nejdříve k dalšímu zákazníkovi a tam počká na „otevření“ okna. Podrobnější rozbor těchto situací lze najít v práci (Fábry, 2006a). Jistě existují i další možnosti jako přerušení jízdy na trase mezi zákazníky, pomalejší přejezd apod. Tato otázka hraje významnou roli především u dynamických úloh (viz podkapitoly 2.6, 3.4 a část 4.2.1).

2.4 Otevřená úloha obchodního cestujícího

Takto se nazývá úloha, v níž se vozidlo nemusí vrátit do výchozího místa a zůstává u posledního navštíveného zákazníka. Toto místo se pak může stát výchozím místem pro následnou optimalizaci (např. v dalším pracovním dnu). Lze snadno změnit matematický model pro řešení této úlohy, mnohem jednodušší je však upravit matici nejkratších vzdáleností tak, že vynulujeme její první sloupec. Pak lze použít standardní model (2.1) – (2.6), tzn. opět se hledá nejkratší Hamiltonův cyklus. Návrat do výchozího místa se ve skutečnosti neprovede, neboli vozidlo zakončí svoji trasu v místě, z něhož se má podle výsledku pokračovat do výchozího místa (s nulovou délkou přejezdu).

2.5 Úloha s více obchodními cestujícími

V případě časových oken či dalších časových omezení (např. časového limitu pro obsloužení všech zákazníků) je většinou nutné naplánovat více okruhů, tedy využít více obchodních cestujících (mTSP)[12]. Bektas (2006) uvádí, že lze úlohy tohoto typu klasifikovat podle několika hledisek. Především se jedná o informaci, zda je v úloze zavedeno pouze jedno nebo několik

[12] Multiple Travelling Salesman Problem. Někdy se pro tuto úlohu používají zkratky m-TSP nebo MTSP.

výchozích míst. Samozřejmě následuje otázka počtu vozidel, která jsou k dispozici v jednotlivých výchozích místech. V některých úlohách je tato hodnota předem dána, jindy je v matematickém modelu počet vozidel proměnnou s horní mezí. Výsledek optimalizace pak určuje, kolik vozidel má být v daném místě připraveno. Protože je použití vozidla kromě variabilních nákladů spojeno s fixními náklady (např. mzda řidiče či částka za pronájem vozidla), může se variabilní počet vozidel stát součástí minimalizační nákladové funkce. Významnou otázkou je také možnost využití vozidel s různou kapacitou. V úloze se mohou vyskytnout některá speciální omezení týkající se maximálního počtu zákazníků, které může jedno vozidlo navštívit, maximální vzdálenosti ujeté vozidlem, nutnost vrátit se do stejného výchozího místa apod.

V literatuře lze najít nepřeberné množství reálných aplikací, v nichž je nutné použít více vozidel. Jednou z nich je svoz vkladů z poboček do centrální banky (Svestka a Huckfeldt, 1973). Další dvě publikované aplikace (Lenstra a Rinnooy Kan, 1975 in Bektas, 2006) se týkají tras technických týmů zajišťujících funkčnost v severním Holandsku a obsluhy 200 míst v Utrechtu. V obou případech bylo cílem minimalizovat počet použitých vozidel. Jinou aplikací tohoto typu byl případ rozvržení jízd fotografů, kteří měli za úkol navštívit základní a střední školy (Zhang,1999 in Bektas, 2006). Ve většině úloh je cílem rozhodnout, kolik vozidel nasadit na zajištění požadavků tak, aby jejich počet byl minimální, stejně tak jako celková délka všech uskutečněných tras. Vzhledem k tomu, že se k základní formulaci úlohy často přidávají i specifické podmínky, ať již v podobě časových oken nebo požadavků na rychlost obsluhy aj., stávají se podobné úlohy stále žádanějšími adepty na úspěšné vyřešení. Jako poslední příklad uveďme problém noční bezpečnostní služby (Calvo a Cordone, 2003), spočívající v provádění pravidelných kontrol předem určených objektů.

2.5.1 Úloha s jedním výchozím místem

Pokud předpokládáme použití K vozidel, která jsou připravena v jednom výchozím místě, lze úlohu řešit aplikací matematického modelu, který je založen na Miller-Tucker-Zemlinově formulaci úlohy obchodního cestujícího (Bektas, 2006):

$$\text{minimalizovat} \quad z = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \,, \tag{2.18}$$

za podmínek

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 2,3,...,n, \tag{2.19}$$

$$\sum_{i=1}^{n} x_{ij} = 1, \quad j = 2,3,...,n, \tag{2.20}$$

$$\sum_{j=2}^{n} x_{1j} = K, \tag{2.21}$$

$$\sum_{j=2}^{n} x_{j1} = K, \tag{2.22}$$

$$u_i - u_j + p x_{ij} \leq p - 1, \quad i = 1,2,...,n, \quad j = 2,3,...,n, \quad i \neq j, \tag{2.23}$$

$$x_{ii} = 0, \quad i = 1,2,...,n, \tag{2.24}$$

$$x_{ij} \in \{0,1\}, \quad i, j = 1,2,...,n, \tag{2.25}$$

$$u_i \in R_0^+, \quad i = 1,2,...,n, \tag{2.26}$$

kde K je počet vozidel, n je počet míst, která vozidla musí projet (včetně výchozího místa označeného indexem 1) a c_{ij} představuje nejkratší vzdálenost mezi místy i a j. Bivalentní proměnná x_{ij} nabývá hodnoty 1 v případě, že vozidlo jede do místa j z místa i, hodnoty 0 v opačném případě. Účelová funkce (2.18) představuje celkovou délku všech tras ujetých všemi vozidly. Rovnice (2.19) a (2.20) zajišťují, že každé místo je navštíveno právě jednou, podmínky (2.21) a (2.22) určují, že z výchozího místa vyjede přesně K vozidel a stejný počet vozidel se opět do výchozího místa vrátí. Podmínky (2.24) zamezí u každého místa vytváření smyček.

Soustava podmínek (2.23) s nezápornými proměnnými u_i zabrání vytváření parciálních cyklů, konstanta p určuje maximální počet zákazníků, které může obsloužit jedno vozidlo (Miller et al., 1960). Pokud není parametr p znám, jinou možností, jak zamezit vytváření parciálních cyklů[13], je použít následující soustavu omezujících podmínek (Gavish, 1976):

$$u_i - u_j + (n - K) x_{ij} \leq n - K - 1, \quad i = 1,2,...,n, \quad j = 2,3,...,n, \quad i \neq j. \tag{2.27}$$

Bektas (2006) uvádí formulace modelů autorů Laporte a Nobert (1980) pro nesymetrickou a symetrickou úlohu, které jsou založeny na výše zmíněných podmínkách autorů Dantzig

[13] Vytvářením parciálních cyklů u tohoto typu úloh rozumíme, že žádné vozidlo nesmí uskutečnit více než jeden okruh. V takto definovaném matematickém modelu je tedy počet tras roven přesně počtu vozidel, tj. hodnotě K.

et al. (1954). Christofides et al. (1981) formuloval model pro rozvozní úlohu, založenou na modelu tokové úlohy s proměnnými se třemi indexy. V práci Fábry (2006a) je uveden matematický model této rozvozní úlohy upravený pro mTSP, v němž jsou vynechány podmínky pro kapacitu vozidel.

Pokud jsou zavedeny doby přejezdů mezi jednotlivými místy, případně časová okna a doby, které stráví vozidla v místech, lze řešit zajímavou modifikaci úlohy mTSP, v níž je nutné zákazníky obsloužit v minimálním čase. Jestliže je předem známá doba, za kterou je nutné všechny zákazníky obsloužit, pak je možné minimalizovat celkovou vzdálenost ujetou všemi vozidly, případně minimalizovat počet vozidel nutných ke splnění daného cíle.

2.5.2 Úloha s několika výchozími místy

Speciální modifikací úlohy mTSP je okružní problém, v němž vozidla nevyjíždějí ze stejného výchozího místa, ale existuje více stanovišť, ze kterých vozidla mohou, avšak nemusejí vyjíždět. Otázkou je, zda se vozidlo, které vyjede z některého místa, musí vrátit do téhož stanoviště nebo může ukončit svoji trasu v jiném stanovišti. Protože tato úloha je komplikovanější verzí, a to jak z praktického, tak z teoretického hlediska[14], budeme v následujícím modelu předpokládat, že v každém stanovišti je připraveno jedno vozidlo, které se musí, pokud vyjede, vrátit do tohoto místa. Nechť existuje K stanovišť a n zákazníků. Zavedeme bivalentní proměnnou x_{ij} nabývající hodnoty 1 v případě, že některé vozidlo pojede přímo z místa i do místa j, hodnoty 0 v opačném případě. Existuje tedy celkem $K + n$ míst, indexy 1, 2, …, K jsou zvoleny pro výchozí místa, indexy $K + 1$, $K + 2$, …, $K+n$ pro sídla zákazníků. Matematický model má následující podobu:

$$\text{minimalizovat} \quad z = \sum_{i=1}^{K+n} \sum_{j=1}^{K+n} c_{ij} x_{ij}\,, \tag{2.28}$$

za podmínek

$$\sum_{i=1}^{K+n} x_{ij} = 1, \quad j = K+1, K+2, \ldots, K+n, \tag{2.29}$$

$$\sum_{i=1}^{K+n} x_{ji} = 1, \quad j = K+1, K+2, \ldots, K+n, \tag{2.30}$$

[14] Viz dále rozvozní úlohy (Kapitola 3), u nichž je možnost zakončení trasy v jiném depotu z logistického hlediska mnohem zajímavější než u mTSP.

$$\sum_{j=K+1}^{K+n} x_{ij} \le 1, \quad i = 1,2,...,K, \tag{2.31}$$

$$\sum_{j=K+1}^{K+n} x_{ij} = \sum_{j=K+1}^{K+n} x_{ji}, \quad i = 1,2,...,K, \tag{2.32}$$

$$u_i - u_j + n x_{ij} \le n-1, \quad i = 1,2,...,K+n,\ j = K+1, K+2,...,K+n, \quad i \ne j, \tag{2.33}$$

$$x_{ij} = 0, \quad i, j = 1,2,...,K, \tag{2.34}$$

$$x_{ii} = 0, \quad i = K+1, K+2,...,K+n, \tag{2.35}$$

$$x_{ij} \in \{0,1\}, \quad i, j = 1,2,...,K+n, \tag{2.36}$$

$$u_i \in R_0^+, \quad i = K+1, K+2,...,K+n. \tag{2.37}$$

Účelová funkce (2.28) představuje celkovou vzdálenost, kterou ujedou všechna vozidla. Soustavy rovnic (2.29) a (2.30) zajišťují, že každý zákazník bude navštíven právě jednou. Podmínky (2.31) určují, že z každého výchozího místa vyjede maximálně jedno vozidlo. Rovnice (2.32) zaručují, že počet vozidel vyjíždějících z každého výchozího místa je roven počtu vozidel, které se do tohoto místa vracejí, v našem případě je tento počet roven 0 nebo 1. Soustava nerovností (2.33) tvoří smyčkové podmínky. Soustava rovnic (2.34) zakazuje pohyb vozidel mezi jednotlivými výchozími místy navzájem, podmínky (2.35) zabraňují u každého zákazníka vytváření smyček.

2.6 Dynamická úloha obchodního cestujícího

Předchozí části této kapitoly se týkaly statické úlohy obchodního cestujícího, v níž předpokládáme znalost všech parametrů před započetím okružní jízdy; především se jedná o apriorní znalost všech zákazníků. V dynamické úloze obchodního cestujícího (DTSP)[15] může kdykoli během realizace jízdy přijít nový požadavek od zákazníka nacházejícího se kdekoli v předem ohraničeném území, resp. od kohokoli z předem známé množiny potenciálních zákazníků. Nový požadavek je pak zařazen do předem naplánovaného okruhu, resp. zbývající části okruhu, kterou vozidlo ještě musí absolvovat.

V práci Fábry (2006b) jsou popsány dva základní přístupy k zařazení nově vzniklých požadavků do předem naplánované trasy vozidla: re-optimalizace a použití vkládacího algoritmu. Při re-optimalizaci je nový zákazník zařazen do množiny dosud nenavštívených zákazníků a následně je nalezena optimální trasa s využitím upraveného modelu pro TSP. Nevýhodou

[15] Dynamic Travelling Salesman Problem.

této metody je fakt, že při vyšším počtu dosud nenavštívených míst může optimalizace trvat příliš dlouho, což může být zvláště v případě těchto úloh klíčový problém. Také vysoká frekvence příchodu nových požadavků zvyšuje časovou náročnost opakovaného výpočtu optimálních tras. Tento přístup využili Bell et al. (1983) , Hill et al. (1988), Brown et al. (1987), Psaraftis et al. (1980, 1983), Powell et al. (1988), Dial (1995) a další.

Při použití heuristického vkládacího algoritmu není zaručeno získání optimální trasy, k jeho výhodám ovšem patří rychlost výpočtu. Nově příchozí zákazník je zařazen do plánované trasy mezi dva po sobě následující zákazníky, kteří mají být vozidlem na trase navštíveni. Pro nalezení nejvhodnější dvojice zákazníků lze použít jako kritérium hodnotu prodloužení stávající trasy po zařazení nového zákazníka, která se bude minimalizovat:

$$\Delta z = c_{i_k, n+1} + c_{n+1, i_l} - c_{i_k, i_l}, \tag{2.38}$$

kde i_k a i_l jsou indexy všech bezprostředně po sobě následujících zákazníků, které má vozidlo navštívit podle plánované trasy, a $n+1$ je index nového zákazníka. Vzhledem k tomu, že je možné určit na trase místo, kam má být nový zákazník vložen, prakticky okamžitě i při větším počtu zákazníků, lze tento postup aplikovat především v těch situacích, v nichž hraje významnou roli čas a firma se musí rozhodnout ihned po přijetí nového požadavku (Lund et al., 1996).

U konkrétního typu úlohy lze kombinovat oba přístupy. Pokud je v daný okamžik dostatečná doba pro re-optimalizaci, použije se tento sofistikovaný postup. Jestliže je naopak nutné rozhodnout o zařazení nového zákazníka okamžitě, použije se heuristický algoritmus. Nabízí se samozřejmě možnost využít vynucenou re-optimalizaci po několika aplikacích vkládací metody, a to i za cenu určitého zdržení. Jsou-li v úloze zavedena časová okna a vozidlo plánovaně čeká na trase, lze tuto prodlevu využít právě k re-optimalizaci, nehledě na to, že použití vkládacího algoritmu v úloze s časovými okny DTSPTW[16] (viz část 2.6.4) není efektivní a velice často zařazení zákazníků tímto postupem není možné a v případě potřeby je nutné využít jiné přibližné postupy.

Uvedené metody předpokládají dynamický, navíc i stochastický charakter vzniku požadavků, neuvažují však možnost předpovídat vlastnosti požadavku na základě pravděpodobnosti jeho vzniku z hlediska časového ani prostorového (viz podkapitola 2.6.5).

[16] Dynamic Travelling Salesman Problem with Time Windows.

2.6.1 Re-optimalizace DTSP

Při re-optimalizaci se nejedná o klasickou úlohu obchodního cestujícího, neboť cílem není nalézt nejkratší okruh, ale nejkratší cestu ze stávajícího uzlu do výchozího uzlu č. 1. Následující model striktně předpokládá, že re-optimalizovaná trasa, resp. její dosud neuskutečněná část, začíná u zákazníka, k němuž vozidlo směřuje v okamžiku příchodu nového požadavku:

$$\text{minimalizovat} \quad z = \sum_{i \in U_N} \sum_{j \in U_N} c_{ij} x_{ij}\,, \tag{2.39}$$

za podmínek

$$\sum_{\substack{j \in U_N \\ j \neq j_{next}}} x_{ij} = 1, \quad i \in U_N - \{1\}, \tag{2.40}$$

$$\sum_{\substack{i \in U_N \\ i \neq 1}} x_{ij} = 1, \quad j \in U_N - \{ j_{next} \}, \tag{2.41}$$

$$u_i - u_j + |U_N|\, x_{ij} \leq |U_N| - 1, \quad i \in U_N, \quad j \in U_N - \{ j_{next} \}, \tag{2.42}$$

$$x_{ij} \in \{0,1\}, \quad i, j \in U_N, \tag{2.43}$$

$$u_i \in R_0^+, \quad i \in U_N, \tag{2.44}$$

kde U_N je množina míst, která musí vozidlo ještě navštívit[17], $|U_N|$ je jejich počet. Index $j_{next} \in U_N$ odpovídá místu, k němuž vozidlo směřuje v okamžiku přijetí nového požadavku, a které se tak stává výchozím místem pro následující trasu. Význam jednotlivých podmínek je zřejmý z předchozích částí práce.

Jestliže v původní matici nejkratších vzdáleností se provede změna vzdálenosti $c_{1 j_{next}}$ mezi výchozím místem č. 1 a zákazníkem j_{next}, ke kterému vozidlo právě směřuje, na hodnotu odpovídající délce dosud absolvované trasy[18] z výchozího místa do místa j_{next}, lze pro řešení úlohy použít i modifikovaný matematický model TSP:

$$\text{minimalizovat} \quad z = \sum_{i \in U_N} \sum_{j \in U_N} c_{ij} x_{ij}\,, \tag{2.45}$$

[17] Jako poslední bude navštíveno výchozí místo, tj. uzel č. 1.
[18] V této trase je zahrnutý i zákazník j_{next}.

za podmínek

$$\sum_{j \in U_N} x_{ij} = 1, \quad i \in U_N, \tag{2.46}$$

$$\sum_{i \in U_N} x_{ij} = 1, \quad j \in U_N, \tag{2.47}$$

$$u_i - u_j + |U_N| x_{ij} \leq |U_N| - 1, \quad i \in U_N, \quad j \in U_N - \{1\}, \tag{2.48}$$

$$x_{1 j_{next}} = 1, \tag{2.49}$$

$$x_{ij} \in \{0,1\}, \quad i, j \in U_N, \tag{2.50}$$

$$u_i \in R_0^+, \quad i \in U_N. \tag{2.51}$$

2.6.2 Vkládací algoritmus pro DTSP

Jak bylo uvedeno výše, v reálných úlohách je velmi často nutné se uchýlit k použití přibližných metod, neboť optimalizace selhává z hlediska výpočetní náročnosti, což právě u dynamických úloh hraje klíčovou roli. Vkládací algoritmus je pro dynamickou úlohu obchodního cestujícího vhodným postupem, jehož aplikace poskytuje uspokojivé výsledky prakticky okamžitě.

Nechť $U_N = \{i_1, i_2, ..., i_m\}$ je posloupnost m míst[19] ($i_m = 1$), která mají být vozidlem ještě navštívena podle naplánované trasy. Označíme-li zákazníka s novým požadavkem indexem $r \notin U_N$, hodnotu prodloužení stávající trasy po jeho vložení mezi místa i_k a i_{k+1} lze určit jako

$$\Delta z_k = c_{i_k, r} + c_{r, i_{k+1}} - c_{i_k, i_{k+1}}, \quad k = 1,2,...,m-1. \tag{2.52}$$

Cílem je nalézt takový index t, pro který platí:

$$\Delta z_t = \min_{k=1,2,...,m-1} \Delta z_k . \tag{2.53}$$

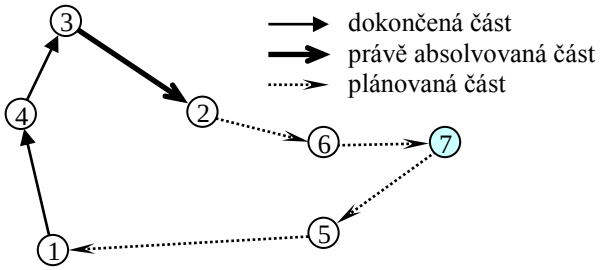
Nový zákazník r bude navštíven bezprostředně po zákazníkovi i_t před zákazníkem i_{t+1}. Zajímavou modifikací tohoto postupu je vkládací algoritmus pro dynamickou úlohu kurýrní služby, v níž jsou do trasy vložena dvě místa, a sice místo vyzvednutí[20] zásilky a místo jejího doručení[21] (viz část 4.2.1.2).

[19] Tato posloupnost může tvořit původní optimální trasu stanovenou před zahájením jízdy vozidla, případně trasu, která byla určena z této optimální trasy postupným vkládáním několika nových požadavků.
[20] Pickup.
[21] Delivery.

2.6.3 Příchod nových požadavků v DTSP

V předchozím textu jsme předpokládali, před každou úpravou trasy, příchod jediného požadavku. Navíc jedinou informací, z hlediska polohy vozidla na trase, byla identifikace zákazníka, ke kterému vozidlo právě směřuje. Pro další úvahy rozdělme trasu vozidla (obr. 2.1) na tři části (Ichoua et al., 2000):

Dokončená část. Jedná se o zákazníky, kteří již byli vozidlem obslouženi. Na obr. 2.1 tato část začíná výchozím místem č. 1 a končí posledním navštíveným zákazníkem č. 3.
Právě absolvovaná část. Tento úsek trasy odpovídá přejezdu mezi posledním navštíveným zákazníkem č. 3 a bezprostředně následujícím zákazníkem č. 2.
Plánovaná část. Před tím, než vozidlo dojede k bezprostředně následujícímu zákazníkovi č. 2, lze přijímat nové zákazníky a provádět jejich zařazení do zbývající časti trasy. V tomto případě bude nový zákazník (uzel č. 7) zařazen mezi všechny dosud nenavštívené zákazníky a určí se nová trasa, která začíná u zákazníka č. 2 a končí ve výchozím místě č. 1.



Obr. 2.1 – Trasa v okamžiku zařazení nového požadavku

Aby zařazení nového zákazníka bylo provedeno co nejefektivněji, a to při použití jakéhokoli postupu, je nutné znát přesnou lokalitu zákazníka a přesnou polohu vozidla. Oba tyto údaje v dnešní době technologií GPS je možné určit s dostatečnou přesností, a tudíž nepředstavují pro řešení úlohy žádný problém. Pokud bychom chtěli tyto údaje co nejefektivněji využít, musíme se důkladně zabývat znalostí nejkratších vzdáleností mezi jednotlivými místy.

Pro optimalizaci statické úlohy TSP je dána matice nejkratších vzdáleností pro předem známá místa. Při příchodu nového požadavku je nutné určit nejkratší vzdálenosti mezi novou lokalitou a všemi dosud nenavštívenými místy. I tento krok může sám o sobě znamenat časově náročnou úlohu (z hlediska rychlosti, s jakou je nutné u DTSP reagovat na nově vzniklé požadavky). Existují tři základní možnosti, jak nejkratší vzdálenosti určit:

1) Pokud se nová lokalita přímo nachází v databázi míst, k nimž existuje matice nejkratších vzdáleností (tj. u statické úlohy TSP byly nejkratší vzdálenosti vyhledány v této matici jen pro existující místa), pak je otázka získání nejkratších vzdáleností vyřešena prakticky okamžitě[22]. Toto je samozřejmě hypotetická možnost, nikoli však pro reálné aplikace, v nichž jsou databáze míst, tj. zákazníků (i těch, jejichž požadavky vznikají až v průběhu realizace trasy) předem známy[23].

2) Je možné určit nejbližší místo (např. na základě GPS souřadnic), které se v databázi míst nachází, a pracovat s určitou aproximací reálných vzdáleností týkající se nové lokality s využitím nejbližší lokality v databázi.

3) Univerzálním a přesným postupem je samozřejmě zjistit nejkratší vzdálenosti pomocí existujících dat s použitím optimalizačních algoritmů. Vzhledem k tomu, že podobné informace dnes nabízejí i některé internetové servery, je využití této možnosti jen otázkou finančních prostředků a nezbytných technologických změn.

Protože v DTSP je důležitá nejen nově zařazovaná lokalita, ale i aktuální poloha vozidla, je nutné se zabývat i možností změny právě absolvované části trasy vozidla (viz obr. 2.1), kterou jsme doposud nepřipouštěli. V reálné situaci se totiž může stát, že lokalita příchozího požadavku se nachází v těsné blízkosti místa, v níž se právě pohybuje vozidlo. Je evidentní, že striktní dodržení podmínky navštívit plánované místo je v tomto případě nesmyslné, neboť budoucí návrat vozidla k obsloužení nového požadavku může představovat významné časové zdržení i vynaložení nemalých finančních prostředků.

Řešení vychází z výše uvedených úvah o získání nejkratších vzdáleností mezi novou lokalitou a všemi dosud nezařazenými místy. Ke všem těmto místům přibyde aktuální poloha vozidla v okamžiku příchodu nového požadavku. Aby byl výpočet ještě přesnější, bylo by nutné využít odhad doby výpočtu nové trasy vozidla, jeho směru a rychlosti. Jako aktuální polohu totiž, v případě delšího výpočtu,[24] musíme stanovit místo, v němž se bude vozidlo nacházet za několik okamžiků. Nezbytnost zanesení těchto zpřesňujících úvah do řešení úlohy závisí na konkrétním případu a na jejich přínosu k celkovému efektu optimalizace trasy.

[22] Pojem „okamžitě“ je samozřejmě relativní a jeho význam závisí na rozsahu databáze míst a rychlosti prohledávacích algoritmů vhledem k času, který je pro výpočet určen.
[23] Jde o dynamickou úlohu v tom smyslu, že sice předem známe všechny potenciální zákazníky, ale není předem známo, od kterého z nich přijde požadavek.
[24] Zjištění nejkratších vzdáleností a následná re-optimalizace.

Příchod nového požadavku ještě automaticky neznamená jeho zahrnutí do stávající trasy. Pokud by zařazením požadavku došlo k překročení pracovní doby, je nutné realizovat jiné řešení v podobě výjezdu dalšího vozidla či přesunutí zpracování požadavku na následující pracovní den. V každém případě v dynamických úlohách, kromě vzdálenosti, hraje důležitou roli čas, což je samo o sobě zřejmé z názvu tohoto typu úloh.

2.6.4 DTSP s časovými okny

Stejně jako ve statické úloze obchodního cestujícího, také v její dynamické verzi je často u každého zákazníka zadán časový interval, ve kterém má být realizována obsluha. V dynamické úloze obchodního cestujícího s časovými okny (DTSPTW)[25], na rozdíl od statické úlohy, jsou některé požadavky známy předem, zatímco další požadavky se objevují až při samotné realizaci okružní jízdy. Pokud chceme určit, jak dlouho je vozidlo na trase, musíme kromě plánované doby přejezdu t_{ij} mezi místy *i* a *j* sledovat i dobu čekaní před zahájením obsluhy (označme ji W_i pro *i*-tého zákazníka). Pokud vozidlo přijede k zákazníkovi před nejdříve možným termínem zahájení obsluhy, označeném v podkapitole 2.3 jako e_i, pak je hodnota proměnné W_i kladná, v opačném případě je nulová. Označíme-li počet předem známých míst (včetně výchozího místa) jako n_Z, pak celková doba jízdy vozidla je dána následující funkcí:

$$T_Z = \sum_{i=1}^{n_Z}\sum_{j=1}^{n_Z} t_{ij}x_{ij} + \sum_{j=2}^{n_Z} W_j \,. \tag{2.54}$$

Ve statické úloze TSP lze minimalizovat tuto funkci či standardní funkci představující celkovou délku trasy. V takovém případě hodnota (2.54) nesmí překročit přípustnou pracovní dobu. Úplný matematický model této úlohy lze najít v práci Fábry (2006b).

V následujícím textu se zaměříme na úlohu DTSPTW a na její řešení pomocí re-optimalizace. Po příchodu nového požadavku je vytvořen re-optimalizační model, a pokud je nalezeno přípustné řešení, splňující všechna časová okna dosud nenavštívených zákazníků a zároveň přípustnou pracovní dobu, dojde k zařazení nového zákazníka a ke změně plánované trasy. V následujícím modelu je minimalizována doba, kterou stráví vozidlo na trase po případném zařazení zákazníka:

[25] Dynamic Travelling Salesman Problem with Time Windows.

$$\text{minimalizovat} \quad z = \sum_{i \in U_N} \sum_{j \in U_N} t_{ij} x_{ij} + \sum_{\substack{j \in U_N \\ j \neq 1}} W_j \,, \tag{2.55}$$

za podmínek

$$\sum_{j \in U_N} x_{ij} = 1, \quad i \in U_N, \tag{2.56}$$

$$\sum_{i \in U_N} x_{ij} = 1, \quad j \in U_N, \tag{2.57}$$

$$e_i \leq \tau_i \leq l_i, \quad i \in U_N - \{1\}, \tag{2.58}$$

$$\tau_i + t_{ij} - M(1 - x_{ij}) + W_j + v_{ij} = \tau_j, \quad i \in U_N, \quad j \in U_N - \{1\}, \quad i \neq j, \tag{2.59}$$

$$0 \leq v_{ij} \leq M(1 - x_{ij}), \quad i \in U_N, \quad j \in U_N - \{1\}, \quad i \neq j, \tag{2.60}$$

$$\tau_1 = 0, \tag{2.61}$$

$$\tau_j \in R_0^+, \quad j \in U_N - \{1\}, \tag{2.62}$$

$$x_{1 j_{next}} = 1, \tag{2.63}$$

$$x_{ij} \in \{0,1\}, \quad v_{ij} \in R_0^+, \quad i, j \in U_N, \tag{2.64}$$

kde U_N je množina míst (včetně sídla nově přidaného zákazníka), která mají být vozidlem ještě navštívena[26], $|U_N|$ je jejich počet. Index j_{next} odpovídá zákazníkovi, k němuž vozidlo směřuje v okamžiku přijetí nového požadavku. Protože vozidlo nemůže změnit směr jízdy, musí být splněna podmínka (2.63). Před optimalizací je upraven původní čas přejezdu $t_{1 j_{next}}$ na hodnotu odpovídající době trvání jízdy vozidla z výchozího místa k zákazníkovi j_{next}. Proměnné v_{ij} zavedené v rovnicích (2.59) musí respektovat omezení (2.60). Pokud vozidlo jede z místa *i* do místa *j* ($x_{ij} = 1$), pak nutně musí platit $v_{ij} = 0$. V opačném případě ($x_{ij} = 0$) proměnná v_{ij} funguje jako pomocná proměnná zajišťující přípustnost řešení vzhledem k časovému rozvrhu. Kdyby tyto proměnné nebyly v rovnicích (2.59) zavedeny, nebylo by možné zajistit splnění všech uvedených rovnic zároveň. Jestliže v praktické úloze sledujeme pracovní dobu, pak je nutné přidat omezení, které zakazuje účelové funkci (2.55) tuto hodnotu překročit.

Pro úplnost uveďme aplikaci vkládacího algoritmu pro DTSPTW, a to i přesto, že pro okružní a rozvozní úlohy s časovými okny, zvláště pro jejich dynamická rozšíření, není tato me-

[26] Jako poslední bude navštíveno výchozí místo č. 1.

toda příliš vhodná. Často totiž tento postup nevede k nalezení přípustného řešení. Uvádíme jej spíše jako jednu z možností řešení v případě, že optimalizační postup selhává z hlediska přípustného výpočetního času.

Nechť $U_N = \{i_1, i_2, ..., i_m\}$ je posloupnost m míst ($i_m = 1$), která vozidlo má ještě navštívit podle plánované trasy. Označme nového zákazníka indexem $r \notin U_N$ a místa, mezi která jej vložíme, indexy i_k a i_{k+1}. Okamžik obsluhy zákazníka r lze určit jako

$$\tau_r = \max(\tau_{i_k} + t_{i_k r}, e_r), \tag{2.65}$$

přičemž musí platit:

$$e_r \le \tau_r \le l_r. \tag{2.66}$$

Hodnota čekání na obsluhu zákazníka *r* je pak definována jako

$$W_r = \max(0, e_r - \tau_{i_k} - t_{i_k r}). \tag{2.67}$$

Vzhledem k tomu, že vložením nového zákazníka dojde ke změně zbývající části trasy, je zapotřebí přepočítat hodnoty τ_i a W_i pro všechna dosud nenavštívená místa, počínaje tím, před které byl nový zákazník vložen:

$$\tau_{i_{k+1}} = \max(\tau_r + t_{r i_{k+1}}, e_{i_{k+1}}), \tag{2.68}$$

$$W_{i_{k+1}} = \max(0, e_{i_{k+1}} - \tau_r - t_{r i_{k+1}}), \tag{2.69}$$

$$\tau_{i_s} = \max(\tau_{i_{s-1}} + t_{i_{s-1} i_s}, e_{i_s}), \quad s = k+2, k+3, ..., m, \tag{2.70}$$

$$W_{i_s} = \max(0, e_{i_s} - \tau_{i_{s-1}} - t_{i_{s-1} i_s}), \quad s = k+2, k+3, ..., m. \tag{2.71}$$

Při použití uvedeného postupu je nutné hlídat přípustnost upravené trasy z hlediska časových oken, tj. musí platit:

$$e_{i_s} \le \tau_{i_s} \le l_{i_s}, \quad s = k+1, k+2, ..., m. \tag{2.72}$$

Dobu trvání nové trasy po vložení zákazníka *r* mezi místa i_k a i_{k+1} lze psát jako

$$z_k = \sum_{s=1}^{k-1} (t_{i_s i_{s+1}} + W_{i_{s+1}}) + (t_{i_k r} + W_r) + (t_{r i_{k+1}} + W_{i_{k+1}}) + \sum_{s=k+1}^{m-1} (t_{i_s i_{s+1}} + W_{i_{s+1}}). \tag{2.73}$$

Cílem je nalézt takové *v*, pro které platí:

$$z_v = \min_{k=1,2,...,m-1} z_k. \tag{2.74}$$

Nového zákazníka pak vložíme mezi místa i_v a i_{v+1}.

Podobně jako u statické úlohy TSPTW, lze i u DTSPTW definovat „soft“ časová okna a zavést penále, které bude dopravce platit za jejich nedodržení. Nákladová funkce, navržená pro statickou úlohu Larsenem et al. (2004), je rozšířena pro dynamické úlohy, a to jak pro re-optimalizaci, tak pro aplikaci vkládacího algoritmu, v práci Fábry (2006a).

2.6.5 DTSP s apriorní informací

Tato část práce je ve zkrácené verzi převzata z publikace Fábry (2006a). V dynamických okružních úlohách je vznik nových požadavků stochastický, a to jak z hlediska místa, tak z hlediska času. Máme-li k dispozici typ a parametry pravděpodobnostního rozdělení náhodných veličin vyskytujících se v modelu, můžeme významně zvýšit kvalitu nalezeného řešení. Larsen et al. (2004) se zabývají dynamickou úlohou obchodního cestujícího s časovými okny s apriorní informací (ADTSPTW)[27] o pravděpodobnosti vzniku požadavku v určité oblasti. Počet nově příchozích požadavků v určité předem vymezené oblasti se řídí Poissonovým rozdělením se známou intenzitou, okamžik přijetí nového požadavku je hodnotou náhodné veličiny, jejíž pravděpodobnostní rozdělení není známo. V dané práci je uvažována reakční doba po obdržení požadavku, tj. časový interval nutný pro zpracování nového požadavku a rozhodnutí o další strategii vozidla. Pokud neexistuje apriorní informace o intenzitách vzniku požadavku v jednotlivých oblastech, mohou v okamžiku, kdy je ukončena obsluha určitého zákazníka, nastat tři situace:

1) Okno dalšího zákazníka v pořadí je již otevřené nebo bude otevřené před tím, než vozidlo přijede k tomuto zákazníkovi. Vozidlo odjíždí k dalšímu zákazníkovi bezprostředně po dokončení obsluhy předchozího zákazníka.

2) Okno dalšího zákazníka se otevře později než v okamžiku, kdy by k němu vozidlo přijelo, pokud by odjelo hned od zákazníka, jehož obsluhu právě ukončilo. Vozidlo setrvává u právě obslouženého zákazníka a čeká na vhodný okamžik odjezdu k dalšímu zákazníkovi, případně na vznik dalšího požadavku.

3) Žádný zákazník nečeká na obsluhu. V tomto případě vozidlo čeká u zákazníka, jehož obsluhu dokončilo, dokud se neobjeví nový požadavek.

Předpokládejme, že v každé z n_{REG} oblastí se nachází právě jedno stanoviště, kam se může vozidlo přesunout v případě čekání na další obsluhu. S využitím apriorní informace

[27] A Priori Dynamic Travelling Salesman Problem with Time Windows.

o intenzitách vzniku požadavků lze modifikovat strategie chování vozidla po dokončení obsluhy pro výše uvedenou druhou, resp. třetí situaci podle následujících kritérií:

(1) nejbližší stanoviště – po obsluze zákazníka vozidlo odjíždí na stanoviště, které je nejblíže pozici, na které se právě nachází,

(2) nejvytíženější stanoviště – po obsluze zákazníka přejíždí vozidlo na stanoviště do regionu s nejvyšší hodnotou intenzity vzniku požadavků,

(3) nejatraktivnější stanoviště – pro přesun vozidla je vybráno stanoviště s nejvyšší hodnotou atraktivnosti, určenou podle následujícího vzorce:

$$AT_j = \lambda_j \Delta T_j, \quad j = 1,2,...,n_{REG}, \tag{2.75}$$

kde λ_j je intenzita vzniku požadavků v *j*-té oblasti a ΔT_j je délka prostoje vozidla v případě, že bylo pro přesun vybráno *j*-té stanoviště, neboli časový interval mezi současným okamžikem a okamžikem, kdy vozidlo bude muset odjet z *j*-tého stanoviště k dalšímu zákazníkovi. Hodnotu AT_j lze jednoduše interpretovat jako očekávaný počet požadavků vzniklých v *j*-tém regionu během intervalu ΔT_j.

Pokud je na základě jednoho ze tří výše uvedených kritérií pro přesun vozidla vybráno *j*-té stanoviště, pak rozhodnutí o skutečném přesunu vozidla či jeho setrvání u právě obslouženého zákazníka lze řídit prostřednictvím tzv. prahu výhodnosti přesunu L_p. Přesun do *j*-tého stanoviště je výhodný v případě, že pravděpodobnost vzniku alespoň jednoho požadavku během intervalu ΔT_j je rovna minimálně hodnotě L_p. Vzhledem k tomu, že počet požadavků *N* vzniklých v *j*-tém regionu během ΔT_j je diskrétní náhodná veličina s Poissonovým rozdělením s intenzitou λ_j, pravděpodobnost vzniku *k* požadavků během tohoto intervalu lze určit jako

$$P\{N = k\} = \frac{1}{k!}(\lambda_j \Delta T_j)^k e^{-\lambda_j \Delta T_j}, \quad k = 0,1,2,... \tag{2.76}$$

Pro pravděpodobnost vzniku alespoň jednoho požadavku lze pak psát

$$P\{N \geq 1\} = 1 - e^{-\lambda_j \Delta T_j}. \tag{2.77}$$

Jak bylo uvedeno, přesun do *j*-tého stanoviště bude uskutečněn v případě, že pravděpodobnost (2.77) bude rovna alespoň prahu výhodnosti přesunu L_p:

$$P\{N \geq 1\} \geq L_p . \tag{2.78}$$

Dosazením (2.78) do (2.77) a využitím vztahu (2.75) dostáváme po jednoduché úpravě výslednou podmínku pro přesun vozidla do *j*-tého stanoviště:

$$AT_j \geq -\ln(1 - L_p) . \tag{2.79}$$

Uvedená podmínka dává do souvislosti atraktivnost vybraného *j*-tého stanoviště a práh výhodnosti přesunu.

Jak ukazují výsledky výpočetních experimentů (Larsen et al., 2004), přesun volného vozidla do atraktivních stanovišť je výhodný z hlediska většího počtu zákazníků, které je možné během dne uspokojit. Použití této strategie je však značně nevýhodné z hlediska celkové vzdálenosti, kterou vozidlo během dne ujede. Nejedná se samozřejmě o větší vzdálenost ujetou v důsledku vyššího počtu zákazníků, ale v důsledku umělých přejezdů vozidla do stanovišť na základě znalosti apriorní informace. Důležitou roli hraje tzv. stupeň dynamiky[28], který charakterizuje konkrétní úlohu:

$$DOD = \frac{n_{IR}}{n_{AR} + n_{IR}} 100\%, \tag{2.80}$$

kde n_{AR} je počet předem známých zákazníků[29] a n_{IR} počet nově vzniklých požadavků[30].

S rostoucím stupněm dynamiky roste i rozdíl mezi celkovou vzdáleností ujetou vozidlem, které přejíždí do atraktivnějších míst a vzdáleností ujetou vozidlem, které čeká u právě obslouženého zákazníka, případně přejíždí do nejbližšího stanoviště. Pokud jde o zpoždění obsluhy oproti nejpozději přípustným začátkům obsluhy, při nízkém stupni dynamiky dochází prakticky ke stejnému zpoždění při všech použitých strategiích[31]. Při vyšších stupních dynamiky je častý přejezd vozidla příčinou mnohem většího zpoždění než v případě, že vozidlo nevyužívá apriorní informaci. Vhodným kompromisem je v tomto případě použití nejbližšího či nejvytíženějšího stanoviště.

[28] Degree of Dynamism.
[29] Advance-Request Customers.
[30] Immediate-Request Customers.
[31] Jako účelová funkce je použita celková hodnota zpoždění, která se minimalizuje.

Obecně lze říci, že dostupnost apriorní informace je výhodná z hlediska vyššího počtu obsloužených zákazníků a s tím souvisejících vyšších tržeb za poskytnuté služby. Nevýhodou je nárůst jak přepravních nákladů, tak i nákladů spojených se zpožděním obsluhy. Volba nejvhodnější strategie závisí na charakteru konkrétní úlohy. Jak bylo uvedeno, důležitou roli pro výběr optimální strategie hraje také stupeň dynamiky konkrétní aplikace. Zajímavý pohled na čekání vozidel v dynamických úlohách nabízejí Branke et al. (2005), kteří odvozují strategie čekaní pro speciální případ se dvěma vozidly.

2.6.6 DTSP s více obchodními cestujícími

V podkapitole 2.4 byla popsána úloha s více obchodními cestujícími ve své statické verzi. Její dynamické rozšíření spolu s jejím řešením je podrobně popsáno v práci Fábry (2006a). Podobně jako u DTSP s jedním obchodním cestujícím, také v případě několika vozidel mohou být nové požadavky zařazovány do plánovaných tras pomocí re-optimalizačního modelu či s použitím některého heuristického postupu, např. modifikovaného vkládacího algoritmu. Opět je zde zachován předpoklad, že můžeme měnit trasy vozidel počínaje místem, do kterého vozidla směřují v okamžiku příchodu nového požadavku.

Jako účelovou funkci lze použít celkovou délku všech tras nebo maximální dobu, kterou stráví všechna vozidla na trase. Všechny uvedené postupy navíc předpokládají, že z každého výchozího místa (pokud jich existuje více), může vyjet maximálně jedno vozidlo. Matematické modely a vkládací algoritmy modifikované pro tento speciální typ úlohy formulované v práci Fábry (2006a) záměrně neuvádím z důvodu omezeného rozsahu této práce, a především pak vzhledem k jejímu zaměření.

2.7 Heuristické algoritmy

Vzhledem k tomu, že úloha obchodního cestujícího a její uvedené modifikace patří mezi NP-obtížné úlohy, je přirozené, že k řešení rozsáhlejších problémů je nutno použít přibližné metody namísto exaktních postupů a matematických modelů (Eiselt a Sandblom, 2000). K těmto metodám se řadí heuristické a metaheuristické algoritmy[32]. Zatímco metaheuristické algoritmy jsou obecně definované algoritmy, které lze aplikovat na většinu úloh celočíselného a smíšeně-celočíselného programování, heuristické algoritmy jsou konkrétní postupy cíleně vyvinuté pro konkrétní úlohy. Pokud jde o úlohu obchodního cestujícího (či rozvozní úlohy), lze rozdělit heuristiky na 3 základní skupiny:

[32] Zkráceně heuristiky a metaheuristiky.

1) Heuristiky generující trasu (construction heuristics). Metody vytvářejí trasu postupným přidáváním jednotlivých míst.

2) Heuristiky zlepšující trasu (improvement heuristics). Postupy, které vycházejí z již vytvořené trasy a jejichž cílem je zlepšit řešení, tj. upravit stávající trasu tak, aby budoucí trasa byla efektivnější.

3) Zatřiďující heuristiky (merge heuristics). Metody vycházejí z nepřípustného řešení, které v případě úlohy obchodního cestujícího obsahuje několik ilegálních (parciálních cyklů). Cílem je tyto cykly postupně zatřídit tak, aby bylo získáno přípustné řešení, tj. jediný (Hamiltonův) cyklus.

Podstatou všech přibližných metod je poskytnutí „kvalitního“ přípustného řešení v přijatelném čase. Základním požadavkem tedy je, aby daný algoritmus byl polynomiální (Papadimitriou a Steiglitz, 1998).

Metoda nejbližšího souseda

Jedná se zřejmě o nejjednodušší postup generující trasu, který je založen na intuitivním přístupu hledání nejbližších míst, která je možné navštívit. Jedná se o tzv. *greedy* metodu. Algoritmus vychází z postupu Rosenkrantze et al. (1977) popsaného Eiseltem a Sandblomem (2000). Nechť U^* je posloupnost míst, odpovídající vygenerované trase, dále i je výchozí uzel, j je poslední uzel na vytvářené trase a z je celková délka trasy.

Krok 1: Inicializace. $U^* = \{i\}$; $j = i$; $z = 0$.

Krok 2: Najdi místo l takové, že $c_{jl} = \min_{v \notin U^*} c_{jv}$. $U^* = U^* + \{l\}$; $z = z + c_{jl}$; $j = l$.

Krok 3: Pokud je $|U^*| < n$, pak jdi na krok 2, jinak jdi na krok 4.

Krok 4: $U^* = U^* + \{i\}$; $z = z + c_{ji}$. Konec.

Jak uvádějí autoři, výpočetní složitost algoritmu je $O(n^2)$. Metoda může být vylepšena jejím použitím s postupným výběrem výchozího uzlu, tedy $i = 1,2,...,n$. V takovém případě je složitost evidentně $O(n^3)$.

Vkládací metoda

Tento algoritmus uvádějí, podle Eiselta a Sandbloma (2000), např. Rosenkrantz et al. (1977), Stewart (1977), Norback a Love (1977) aj. Předpokládejme, že úloha je reprezentová-

na úplným neorientovaným grafem $G = \{U, H\}$ a ohodnocení hran splňuje trojúhelníkové nerovnosti. Nechť $U^* \subseteq U$ je množina míst (uzlů), které jsou zařazeny do trasy, $H^* \subseteq H$ je množina hran[33], tvořící vygenerovanou trasu, dále *i* je výchozí uzel a z je celková délka trasy.

Krok 1: Inicializace.

Najdi libovolný uzel $s \neq i$, např. takový, pro který platí: $c_{is} = \max_{v \neq i} c_{iv}$.

$U^* = \{i, s\}$; $H^* = \{(i,s),(s,i)\}$; $z = c_{is} + c_{si}$.

Krok 2: Pokud $U^* = U$, pak jdi na krok 4.

Krok 3: Najdi $\Delta z = c_{ls} + c_{sk} - c_{lk} = \min_{\substack{v \notin U^* \\ (i,j) \in H^*}} \{c_{iv} + c_{vj} - c_{ij}\}$.

$U^* = U^* + \{s\}$; $H^* = H^* + \{(l,s),(s,k)\} - \{(l,k)\}$; $z = z + \Delta z$.

Jdi na krok 2.

Krok 4: Konec.

Podobně jako metoda nejbližšího souseda, také vkládací metoda může jako výchozí uzel vybrat postupně všechny uzly. Navíc v kroku 3 může být použito jiné kritérium než minimální prodloužení trasy při zařazení daného uzlu, např. nalezení nejbližšího souseda k již zařazeným uzlům, tj. aplikace greedy přístupu. Gendreau et al. (1998a) popisují zobecněný vkládací algoritmus pro úlohu TSPTW.

Metoda výhodnostních čísel (savings)

Algoritmus pochází z roku 1964, kdy ji navrhli autoři Clarke a Wright. Je založený na výpočtu tzv. výhodnostních čísel pro každou dvojici uzlů. Laporte a Semet (2002) tuto metodu uvádějí spolu s jejími modifikacemi ve sborníku vydaném Tothem a Vigem (2002). Základní algoritmus má dvě verze:

Krok 1: Vytvoř $n-1$ cyklů $(1,i,1)$ pro $i = 2,3,...,n$.

Krok 2: Vypočti pro každou dvojici uzlů (různých od uzlu 1) výhodnostní číslo:

$s_{ij} = c_{i1} + c_{1j} - c_{ij}$ $(i, j = 2,3,...,n; i \neq j)$.

Krok 3: Setřiď výhodnostní čísla sestupně.

[33] $H^* \subset H$ platí v úplném grafu pro n > 3.

Paralelní verze.

Krok 4: Najdi nejvyšší hodnotu s_{ij}, kde i a j jsou součástí dvou různých cyklů a zároveň první cyklus obsahuje hranu $(1, j)$ a druhý hranu $(i,1)$. Proveď spojení těchto dvou cyklů odstraněním hran $(1, j)$ a $(i,1)$, a zahrnutím hrany (i, j). Tento krok se opakuje tak dlouho, dokud nevznikne jediný, tedy Hamiltonův cyklus.

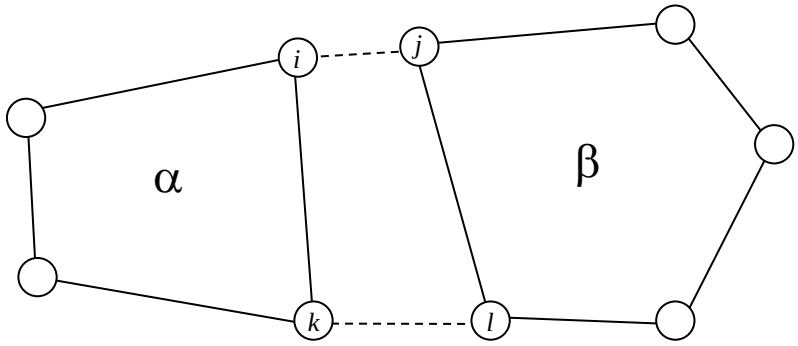
Sekvenční verze.

Krok 4: Uvažujme cyklus $(1,i,...,j,1)$. Najdi nejvyšší hodnotu s_{ki} nebo s_{jl}, kde k a l leží v jiném cyklu, který obsahuje hranu $(k,1)$ nebo $(1,l)$. Proveď spojení cyklů podobně jako v paralelní verzi. Tento krok opakuj tak dlouho, dokud nevznikne Hamiltonův cyklus.

Na metodu výhodnostních čísel lze v jistém smyslu pohlížet jako na metodu zatřiďování, která je založena na stejném principu, ovšem pro spojování cyklů využívá jiné kritérium.

Metoda zatřiďování

Na rozdíl od metody výhodnostních čísel, v níž se zatřiďují (shlukují) vždy dva cykly, které obsahují výchozí místo, tato metoda, kterou uvádí např. Pelikán (2001), spojuje cykly, které nemají společný uzel[34]. Předpokládejme, že existují dva takové cykly (obr. 2.2).



Obr. 2.2 – Zatřiďování parciálních cyklů

Sloučení dvou cyklů α a β v jeden je výhodné, pokud najdeme dvě hrany (i, j) a (k,l), kde $i,k \in \alpha$ a $j,l \in \beta$, takové, že $c_{ij} + c_{kl} < c_{ik} + c_{jl}$. Protože v úloze hledáme Hamiltonův

[34] Tedy pouze jeden cyklus obsahuje výchozí uzel.

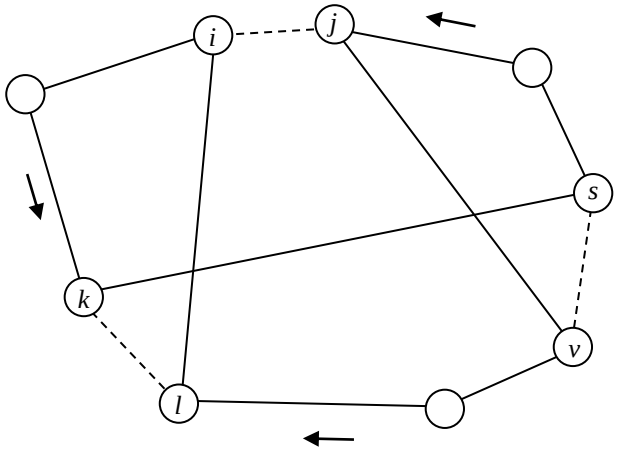
cyklus, tzn. že nakonec musí být všechny parciální cykly sloučeny, pro zatřiďování použijeme následující kritérium:

$$D_{\alpha,\beta} = \min_{\substack{i,k\in\alpha \\ j,l\in\beta}} (c_{ij} + c_{kl} - c_{ik} - c_{jl}) . \tag{2.81}$$

Hledáme takové dva cykly, pro které je hodnota tohoto kritéria minimální[35]. Provedeme sloučení těchto cyklů nahrazením dvou hran jinými dvěma hranami podle (2.81). Postupně dojde ke sloučení všech cyklů[36].

Metoda výměn

Flood (1956) ukázal, že lze často zlepšit řešení nalezené některou z generujících heuristik výměnou dvou neincidentních hran za jiné dvě neincidentní hrany. Lin (1965) formuloval postup, který se označuje jako 3-opt algoritmus a spočívá k výměně tří neincidentních hran za jiné tři hrany v případě, že tato výměna přináší zlepšení z hlediska hodnoty sledovaného kritéria[37]. Výsledkem musí být samozřejmě opět Hamiltonův cyklus. Na obr. 2.3 je znázorněno využití tohoto postupu. Předpokládáme-li neorientovaný graf, vede původní Hamiltonův cyklus přes uzly $i,...,k,s,...,j,v,...,l,i$ (ve směru naznačeném šipkami) nebo samozřejmě ve směru opačném. Pokud nahradíme hrany $(k,s),(j,v)$ a (l,i) hranami[38] (k,l), (v,s) a (j,i), a pokud je toto řešení lepší než řešení původní, přijmeme jej. Jedná se o Hamiltonův cyklus $i,...,k,l,...,v,s,...,j,i$.



Obr. 2.3 – aplikace 3-opt algoritmu

[35] I za cenu prodloužení obou tras.
[36] Výchozí množinu cyklů lze získat například metodou minimálního párování (v případě lichého počtu uzlů bude poslední cyklus obsahovat tři uzly).
[37] V případě výměny dvou hran za jiné dvě hrany, kterou navrhoval Flood, se tedy jedná o 2-opt výměnu.
[38] Hrany jsou vyznačeny čárkovaně.

Je zřejmé, že průjezd vozidla některými hranami se děje ve směru opačném než v původním řešení. Konkrétně se jedná o dvě hrany mezi uzly *l* a *v*. Počet hran, které lze tímto způsobem vyměňovat, může být samozřejmě i vyšší než tři. Pokud označíme jejich počet *r*, pak lze metodu zobecnit a nazývá se *r*-opt. V každé iteraci je možné vybrat *r* hran $\binom{n}{r}$ způsoby a existuje *r*! možností[39], jak spojit vzniklé řetězce (Eiselt a Sandblom, 2000). Proto se v reálných situacích používá $r = 2$ nebo $r = 3$. Vylepšení tohoto přístupu představili Lin a Kernighan (1973) v podobě metody, kde hodnota *r* není předem dána. Cílem je omezit množinu všech *r*-opt výměn tím, že metoda uvažuje jen takové *r*-opt výměny, které jsou vytvářeny z řady speciálně nalezených 2-opt výměn. Namísto hledání jedné zlepšující výměny, algoritmus vytváří posloupnost 2-opt výměn, které dohromady vyústí v získání lepšího řešení. Tento postup byl později modifikován za účelem jeho zefektivnění (např. Helsgaun, 2000).

[39] Ne všechny způsoby jsou samozřejmě přípustné, neboť odstraněné hrany nesmí být incidentní.

3 Rozvozní úlohy

Předchozí kapitola byla věnována okružním úlohám, v nichž mají všechna navštívená místa nulové požadavky. Z tohoto důvodu nebylo nutné v modelech zavádět kapacitu obsluhujících vozidel. Jestliže zákazník požaduje, aby mu bylo z nějakého centrálního distribučního místa dodáno určité množství zboží či materiálu, nebo má být naopak určité množství od zákazníka odvezeno, je zapotřebí takový problém řešit jako rozvozní úlohu (VRP)[40]. Jak bylo uvedeno v Kapitole 1, česká terminologie bohužel tyto dva základní typy aplikací zahrnuje pod výše uvedený společný název, a tudíž se mluví o rozvozní úloze i např. v případě svozu směsného odpadu. Dokonce i matematický model je mnohdy nazýván jako model rozvozní úlohy, i když je zcela evidentní, že se jedná o model „úlohy svozu“. Správně by tedy tato část práce měla nést název „Úlohy rozvozu a svozu“. Cílem práce ovšem není měnit zažitou terminologii, a proto se budu držet tohoto označení.

Desrosiers et al. (1995) uvádějí třídu úloh VRP jako speciální podskupinu úloh třídy Pickup and Delivery Problem (PDP), tedy v českém překladu „úlohy vyzvednutí a doručení“. Konkrétní aplikaci s názvem „zobecněný PDP“ bude věnována podkapitola 4.3. Podobně jako u TSP, lze také v případě VRP najít celou řadu speciálních úloh, kterými se bude zabývat tato kapitola. Pro nalezení optimálního řešení rozvozních úloh lze použít metodu větvení a hranic[41], metodu větvení a řezů[42], případně metodu větvení a oceňování[43]. Tyto metody popisují Toth a Vigo (1998), Achuthan et al. (2003), Desrochers et al. (1992), Kolen et al. (1987) aj. Přibližné řešení poskytují heuristické či metaheuristické postupy, založené většinou na metodách tabu search, genetických algoritmech, metodách mravenčích kolonií apod. Jejich použitím se zabývají Čičková (2005), Potvin a Bengio (1996), Potvin et al. (1996), Bräysy a Gendreau (2005a, 2005b), Gendreau et al. (1994), Russel (1995), Taillard et al. (1997), Golden et al. (1998) aj. Většina uvedených prací se týká rozvozních úloh s časovými okny (VRPTW)[44]. V úloze, kterou popisují Angel et al. (1972), je cílem získat takový rozpis linek školních autobusů, aby počet navržených tras byl minimální. Žádný z autobusů nesmí být přeplněn kvůli bezpečnosti cestujících žáků a v žádném případě nesmí dojít k nedodržení požadovaného rozpisu vyložení žáků na určených místech. Zajímavým problémem je vyrovná-

[40] Vehicle Routing Problem.
[41] Branch-and-Bound Algorithm.
[42] Branch-and-Cut Algorithm.
[43] Branch-and-Price Algorithm. Tento postup je založen na metodě generování sloupců, jejíž použití v celočíselném programování lze mj. najít v práci Fábry (2005b), resp. Fábry a Pelikán (2004).
[44] Vehicle Routing Problem with Time Windows.

vání pracovní zátěže mezi jednotlivými vozidly (Okonjo-Adigwe, 1988). V této úloze jsou přidána další omezení ve formě dolních a horních mezí doby jízdy a celkové hmotnosti jednotlivých vozidel.

Pro zjednodušení budou všechny matematické modely předpokládat, že cílem úlohy je svézt zboží od zákazníků do jednoho, případně několika depotů, budou se tedy týkat úloh svozu. Tento předpoklad nemá vliv na řešení úlohy, je ale třeba jej vzít v potaz při závěrečné interpretaci výsledků analýzy.

3.1 Rozvozní úloha s jedním vozidlem

V této úloze existuje jediné výchozí místo označené č. 1, v němž je umístěno jediné vozidlo o známé kapacitě *V*. Dle výše uvedeného předpokladu má vozidlo zajistit svoz materiálu či zboží od *n*-1 zákazníků, u nichž je známá velikost požadavku $0 < q_i \leq V \quad (i = 2,3,...,n)$. Stejně jako v okružních úlohách uvedených v předchozí kapitole, je i v této kapitole definována matice[45] nejkratších vzdáleností mezi všemi místy; vzdálenost mezi místy i a j je označena $c_{ij} \quad (i, j = 1,2,...,n)$. Matematický model úlohy, v níž je cílem minimalizovat celkovou vzdálenost ujetou vozidlem, splnit požadavky zákazníků a nepřekročit kapacitu vozidla, lze pak zapsat následujícím způsobem (Pelikán, 2001):

$$\text{minimalizovat} \quad z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} x_{ij} \,, \tag{3.1}$$

za podmínek

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 2,3,...,n, \tag{3.2}$$

$$\sum_{i=1}^{n} x_{ij} = 1, \quad j = 2,3,...,n, \tag{3.3}$$

$$u_i + q_j - V(1 - x_{ij}) \leq u_j, \quad i = 1,2,...,n, \quad j = 2,3,...,n, \quad i \neq j, \tag{3.4}$$

$$u_i \leq V, \quad i = 2,3,...,n, \tag{3.5}$$

$$u_1 = 0, \tag{3.6}$$

$$x_{ij} \in \{0,1\}, \quad i, j = 1,2,...,n, \tag{3.7}$$

$$u_i \in R_0^+, \quad i = 1,2,...,n, \tag{3.8}$$

[45] Předpokládejme obecně nesymetrickou matici.

kde x_{ij} je bivalentní proměnná nabývající hodnoty 1 v případě, že vozidlo navštíví místo *j* bezprostředně po návštěvě místa *i*, hodnoty 0 v opačném případě. Účelová funkce (3.1) představuje celkovou vzdálenost ujetou vozidlem. Omezující podmínky (3.2) a (3.3) stanovují, že každé místo (kromě místa výchozího) bude navštíveno právě jednou.

Nerovnosti (3.4) zamezují vytváření parciálních cyklů, tj. cyklů neobsahujících výchozí místo. Zároveň představují bilanci nákladu vozidla, neboť hodnoty proměnných u_i se zvyšují v úzké souvislosti s rostoucí velikostí nákladu na vozidle. Podmínky (3.5) nepovolují překročení kapacity vozidla *V* po návštěvě libovolného zákazníka, rovnice (3.6) zaručuje, že vozidlo bude při každém výjezdu na trasu prázdné.

Je zapotřebí dát pozor na správnou interpretaci výsledných hodnot proměnných u_i. Velmi často se mylně vysvětlují jako velikost nákladu vozidla po návštěvě určitého místa; k tomuto závěru lze dospět jen v případě, že v rámci okruhu dojde k úplnému vyčerpání kapacity vozidla. V opačném případě může pro některá místa ve výsledku platit, že hodnota u_i je vyšší než skutečný náklad vozidla po jeho návštěvě *i*-tého místa. V případě potřeby lze tento „nedostatek“ napravit např. následující změnou účelové funkce:

$$\text{minimalizovat} \quad z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} + \frac{1}{M}\sum_{i=2}^{n} u_i \,, \tag{3.1a}$$

kde *M* je vysoká konstanta. Pak lze již s určitostí tvrdit, že příslušné hodnoty u_i představují náklad vozidla při odjezdu z jednotlivých míst.

Jestliže požadavek některého ze zákazníků převyšuje kapacitu vozidla, je nutné úlohu řešit jako rozvozní problém s dělenou dodávkou (SDVRP)[46], resp. nakládkou. Tato úloha bude popsána v samostatné části této kapitoly. Protože nejsou zadána žádná časová omezení, jedno vozidlo je možné využít opakovaně, tj. po svém návratu do depotu vozidlo vyjede na další trasu, a tento proces se opakuje, dokud nebudou obslouženi všichni zákazníci. Nutnost využít k rozvozu či svozu více vozidel je vysvětlena v následujícím textu.

3.2 Rozvozní úloha s časovými okny

Stejně jako v úloze obchodního cestujícího, také v rozvozní úloze se velmi často zavádějí časová okna, během nichž je nutné navštívit, případně i obsloužit zákazníky. Protože rozvozní

[46] Split Delivery Vehicle Routing Problem.

úlohy, resp. úlohy svozu jsou ve většině případů spojené s vykládkou, resp. nakládkou zboží či materiálu, očekáváme znalost dob trvání těchto operací, které se samozřejmě mohou pro jednotlivé zákazníky lišit. Obdobně jako v úloze obchodního cestujícího, budeme i v rozvozních úlohách pro tyto hodnoty používat značení S_i $(i = 2,3,...,n)$. Dále je nutné znát dobu přejezdu z místa i do místa j, označenou t_{ij} pro $i, j = 1,2,...,n, i \neq j$. V matematickém modelu (3.1) – (3.8) je tedy nutné zavést pro každé místo časovou proměnnou τ_i, udávající okamžik příjezdu vozidla do místa i $(i = 1,2,...,n)$ a přidat soustavu omezujících podmínek (2.12). Na rozdíl od modelu úlohy TSP, v modelu úlohy VRP tato soustava nenahrazuje klasické smyčkové podmínky, neboť podmínky (3.4) mají zcela jiný význam než podmínky (2.4). Je tedy nutné ponechat tyto bilanční podmínky v modelu a podmínky (2.12) tento model pouze rozšiřují. Samozřejmě je nezbytné přidat omezující podmínky (2.13) – (2.15) z důvodu respektování všech časových oken. V úlohách s časovými okny většinou nevystačíme s jedním vozidlem, neboť to nemůže být v extrémním případě na dvou místech zároveň. Proto je zapotřebí pro úspěšnou realizaci rozvozu či svozu zapojit další vozidla (viz následující část kapitoly).

Speciálním případem úloh s časovými okny je úloha, v níž některá místa je nutné navštívit několikrát během určitého časového období. V literatuře je tato úloha známá pod pojmem (Multi-) Period Vehicle Routing Problem (Francis et al., 2008). Časová okna mohou být zadána jako některé (celé) dny v týdnu, přičemž zákazníka je nutné navštívit právě během těchto časových oken, tj. v určité dny. Favaretto et al. (2007) uvádějí matematický model pro úlohu, v níž každý zákazník musí být navštíven několikrát během několika časových oken a délka žádné trasy nesmí překročit zadanou hodnotu. Kromě exaktního modelu, vhodného pouze pro méně rozsáhlé případy, autoři popisují přístup založený na metaheuristickém algoritmu mravenčích kolonií. Golden et al. (2008) uvádějí další varianty této úlohy, např. problém s několika výchozími místy, pro který nabízejí i jednoduchý heuristický algoritmus.

3.3 Rozvozní úloha s více vozidly

Jak bylo uvedeno výše, v okružních úlohách může, resp. musí být k obsluze zákazníků využito více vozidel. Ta mohou být k dispozici v jednom či několika depotech. Pokud v úloze TSP s jedním depotem nejsou zavedena časová okna nebo další podmínky omezující délku cyklu, ať již z hlediska času nebo vzdálenosti, pak je optimálním řešením Hamiltonův cyklus a využito jediné vozidlo. V případě zavedení časových oken je většinou nutné použít několik

vozidel, která mohou vyjíždět z jednoho nebo několika depotů. Více vozidel je nutné využít právě v úlohách VRP, neboť u nich je navíc nutné respektovat kapacitu vozidla.

Jestliže je v úloze VRP dáno jen jedno výchozí místo, a nejsou-li zadána žádná časová omezení, lze řešení úlohy interpretovat dvěma základními způsoby:

1) V depotu je jedno vozidlo, které projede všechny výsledné trasy postupně.
2) V depotu je více vozidel, která mohou vyjet na trasy paralelně.

Jak bylo zmíněno v předešlé části, v případě existence časových oken první možnost prakticky nepřipadá v úvahu a zákazníci jsou obslouženi několika vozidly, která vyjíždějí na trasy paralelně nebo s jistým časovým zpožděním (podle času příjezdu do prvních míst na trasách). Zajímavou modifikací je úloha, v níž je v jednom depotu více vozidel s různou kapacitou.

3.3.1 Vozidla s rozdílnou kapacitou

Předpokládejme, že ve výchozím místě jsou připravena vozidla K typů s kapacitami V_k $(k = 1,2,...,K)$. Matematický model takové úlohy pak lze zapsat následujícím způsobem:

$$\text{minimalizovat} \quad z = \sum_{k=1}^{K} \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}^{k}, \tag{3.9}$$

za podmínek

$$\sum_{k=1}^{K} \sum_{j=1}^{n} x_{ij}^{k} = 1, \quad i = 2,3,...,n, \tag{3.10}$$

$$\sum_{i=1}^{n} x_{ij}^{k} = \sum_{i=1}^{n} x_{ji}^{k}, \quad j = 2,3,...,n, \quad k = 1,2,...K, \tag{3.11}$$

$$u_i + q_j - \overline{V}(1 - x_{ij}^{k}) \le u_j, \quad i = 1,2,...,n, \quad j = 2,3,...,n, \quad i \ne j, \quad k = 1,2,...,K, \tag{3.12}$$

$$q_i \sum_{j=1}^{n} x_{ij}^{k} \le V_k, \quad i = 2,3,...,n, \quad k = 1,2,...,K, \tag{3.13}$$

$$u_1 = 0, \tag{3.14}$$

$$x_{ij}^{k} \in \{0,1\}, \quad i, j = 1,2,...,n, \quad k = 1,2,...,K, \tag{3.15}$$

$$u_i \in R_0^{+}, \quad i = 1,2,...,n. \tag{3.16}$$

V modelu jsou zavedeny proměnné x_{ij}^{k}, jejichž třetí index k představuje pořadí typu vozidla. Účelová funkce (3.9) představuje celkovou vzdálenost, kterou ujedou všechna vozidla, která jsou k rozvozu využita. Rovnice (3.10) zajišťují, že každé místo (kromě místa výchozího)

bude navštíveno právě jednou. Soustava rovnic (3.11) zamezuje tomu, aby z některého místa vyjelo vozidlo jiného typu než to, které do něj vjede. Nerovnosti (3.12) zabraňují vytváření parciálních cyklů, které neobsahují výchozí místo a zároveň vyjadřují bilanci nákladu vozidla. Hodnota $\overline{V}$ je rovna maximu ze všech kapacit vozidel, může být však také vyšší. Aby byla respektována kapacita všech vozidel, musí být splněny podmínky (3.13). Nulovou velikost nákladu ve výchozím místě pro všechny trasy zajišťuje podmínka (3.14).

V reálných úlohách může být omezen počet vozidel daného typu. V případě, že není možné, aby stejné vozidlo absolvovalo po sobě několik tras, je nutné znát disponibilní počet vozidel *k*-tého typu, který označíme p_k. V takovém případě nutnou podmínkou pro existenci přípustného řešení uvedeného matematického modelu jsou následující nerovnosti:

$$\sum_{i=2}^{n} q_i \leq \sum_{k=1}^{K} p_k V_k, \tag{3.17}$$

$$q_i \leq \max_{k=1,2,...,k} V_k = \overline{V}, \quad i = 2,3,...,n. \tag{3.18}$$

Nerovnost (3.17) zajišťuje, že ve výchozím místě jsou vozidla s kapacitami, které jsou dostatečné pro uspokojení požadavků všech zákazníků dohromady. Pokud alespoň jeden požadavek nesplňuje podmínku (3.18), pak jej nelze uspokojit v rámci jednoho okruhu a tudíž není splněna podmínka (3.10), tj. že každý zákazník musí být navštíven právě jednou. V takovém případě musí být na problém nahlíženo jako na úlohu s dělenou dodávkou, resp. nakládkou (viz podkapitola 3.5). Předpokládáme-li omezený vozový park, je samozřejmě nutné do výše uvedeného modelu přidat následující podmínku:

$$\sum_{j=1}^{n} x_{1j}^{k} \leq p_k, \quad k = 1,2,...K, \tag{3.19}$$

která zajistí, že z depotu vyjede maximálně tolik vozidel *k*-tého typu, kolik jich je k dispozici.

V souvislosti s rozdílnou kapacitou vozidel vyvstává otázka, zda je rozumné v účelové funkci používat koeficienty nejkratších vzdáleností. Pro reálné úlohy totiž aproximace finančního hlediska, které je předmětem většiny manažerských úloh, hlediskem vzdálenosti není v případě různě těžkých vozidel s rozdílnou nosností vhodná[47]. Proto by bylo žádoucí zavést

[47] Rozdíl je patrný především v členitém terénu či v městské dopravě.

koeficienty c_{ij}^k, představující náklady[48] na přepravu nákladu mezi místy *i* a *j* vozidlem *k*-tého typu. Účelovou funkci pak můžeme přepsat jako

$$z = \sum_{k=1}^{K} \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij}^k x_{ij}^k. \tag{3.9a}$$

3.3.2 Více výchozích míst

V úlohách s větším počtem vozidel, umístěných v několika výchozích místech, lze teoreticky uvažovat i situace, v nichž se vozidlo nemusí vrátit do výchozího místa, ale může ukončit svou trasu v jiném místě. To by ovšem znamenalo velice důležitou změnu v předpokladech modelu, tedy pokud se jedná o opakované (např. denní či týdenní) plánování tras[49]. Protože v takovém případě v jednom depotu může ukončit svou trasu více vozidel a naopak do některého depotu nepřijede žádné vozidlo, bylo by nutné opustit předpoklad, že v každém stanovišti je právě jedno vozidlo. Jako základní údaje by kromě jiných musely být uvedeny počty vozidel, která jsou k dispozici v každém depotu (pro daný den či týden). Tolik vozidel také maximálně může ze stanoviště vyjet. Samozřejmě musí být v úloze uvažovány i depoty, v nichž se právě nenachází žádné vozidlo, a to z toho důvodu, že může být výhodné, aby některá vozidla v tomto místě svoji trasu ukončila. Samotný model úlohy lze pak poměrně jednoduše upravit tak, že budou zavedeny fiktivní (nulové) vzdálenosti mezi depoty a úlohu budeme dále řešit jako úlohu s cyklickými trasami (tj. s návratem vozidel do místa výjezdu), přičemž případný závěrečný přejezd mezi depoty se prakticky neuskuteční.

Dále budeme předpokládat, že v každém místě je k dispozici právě jedno vozidlo o kapacitě V_k. Laporte et al. (1988), Savelsbergh a Sol (1998) se zabývají úlohami, v nichž mají vozidla stejnou kapacitu. Následující model, v němž budeme uvažovat rozdílnou kapacitu vozidel, je založený na modelu úlohy TSP s několika výchozími místy (2.28) – (2.37) a modelu VRP s několika vozidly *k*-tého typu v jednom výchozím místě (3.9) – (3.16). V první řadě je třeba odlišit *K* výchozích míst a *n* zákazníků. Dále je možné, nikoli však nutné, vozidlům zakázat jejich přejezdy mezi výchozími místy dosazením prohibitivních sazeb $c_{ij} = M \ (i, j = 1,2,...,K)$ nebo vynulováním příslušných proměnných. Matematický model lze pak zapsat následovně:

[48] Ani v tomto případě však není úvaha zcela přesná. Vozidla se shodnou nosností se chovají jinak, pokud jsou na začátku trasy prázdná, a jinak na konci trasy, kdy jsou již plně naložená.
[49] Tato možnost je zajímavá především v situacích, kdy se každý den či týden mění množina zákazníků a také u dynamických rozvozních úloh.

$$\text{minimalizovat} \quad z = \sum_{k=1}^{K} \sum_{i=1}^{K+n} \sum_{j=1}^{K+n} c_{ij} x_{ij}^{k}, \tag{3.20}$$

za podmínek

$$\sum_{k=1}^{K} \sum_{i=1}^{K+n} x_{ij}^{k} = 1, \quad j = K+1, K+2, ..., K+n, \tag{3.21}$$

$$\sum_{i=1}^{K+n} x_{ij}^{k} = \sum_{i=1}^{K+n} x_{ji}^{k}, \quad j = K+1, k+2, ..., K+n, \quad k = 1,2,...K, \tag{3.22}$$

$$\sum_{j=K+1}^{K+n} x_{kj}^{k} \le 1, \quad k = 1,2,..., K, \tag{3.23}$$

$$u_i + q_j - \overline{V}(1 - x_{ij}^{k}) \le u_j, \quad i = 1,2,..., K+n, \quad j = K+1, K+2, ..., K+n, \quad i \ne j, \quad k = 1,2,..., K, \tag{3.24}$$

$$u_i \sum_{j=1}^{K+n} x_{ij}^{k} \le V_k, \quad i = K+1, K+2, ..., K+n, \quad k = 1,2,..., K, \tag{3.25}$$

$$x_{ij}^{k} = 0, \quad i, j = 1,2,..., K, \quad k = 1,2,..., K, \tag{3.26}$$

$$x_{ij}^{k} \in \{0,1\}, \quad i, j = 1,2,..., K+n, \quad k = 1,2,..., K, \tag{3.27}$$

$$u_i = 0, \quad i = 1,2,..., K, \tag{3.28}$$

$$u_i \in R_0^{+}, \quad i = K+1, K+2, ..., K+n. \tag{3.29}$$

Jednotlivé podmínky a jejich význam korespondují s podmínkami uvedenými v předchozích modelech, proto není nutné je podrobněji vysvětlovat. Zvláště v rozvozních úlohách lze namísto minimalizace celkové ujeté vzdálenosti minimalizovat čas potřebný k obsluze všech zákazníků. Typickou aplikací je svoz pošty do třídících center či svoz platebních příkazů z poboček bank do centrálních míst. V těchto úlohách jsou většinou všichni zákazníci předem známí a pro řešení je použito matematického modelu statické rozvozní úlohy, ať již s jedním či několika výchozími místy. V některých případech je předem známá doba, během níž je nutné obsloužit všechny zákazníky a vrátit se zpět do výchozího místa, a přitom ujet minimální vzdálenost.

3.4 Dynamická rozvozní úloha

Dynamické rozšíření rozvozní úlohy naráží na jeden důležitý fakt, a sice na to, že zákazníci mají nenulové požadavky vzhledem ke kapacitě vozidla. Při rozvozu materiálu či jiného homogenního produktu by tedy vozidlo muselo ve výchozím místě naložit více než je součet

požadavků zákazníků, jejichž návštěva byla naplánována při řešení statické úlohy. Tento převis pak může být použit pro uspokojení případných nově příchozích požadavků. S on-line požadavky může být dokonce počítáno i při řešení statické úlohy v podobě umělého snížení kapacity vozidla. Mnohem častější je ovšem případ svozu (např. odpadu), kdy vozidlo může mít na plánované trase určitou rezervu, kterou může využít pro uspokojení nových požadavků. I v tomto případě může být tato rezerva předem uměle navýšena. Speciální úlohou tohoto typu, na které lze využít i pravděpodobnosti vzniku požadavku na určitém předem známém místě, je svoz tříděného odpadu z kontejnerů umístěných na známých stanovištích. Lze provádět jejich pravidelnou kontrolu a požadavek na vyprázdnění vznést až při jejich naplnění. V případě, že známe pravděpodobnosti naplnění kontejneru, je možné vytvářet záměrně rezervu kapacity vozidel, případně i odvézt obsah ne zcela zaplněného kontejneru, pokud je možné tento obsah odhadnout. Tato strategie hraje významnou roli především v případech, kdy distribuční firma zná pravděpodobnostní rozdělení velikosti poptávky, resp. nabídky potenciálních zákazníků.

Na základě dříve uvedených matematických modelů lze poměrně snadno odvodit postupy pro řešení rozvozní úlohy s více vozidly v jednom či více výchozích místech s tím, že z reálného hlediska je nutné znát konkrétní situaci a dokázat odhadnout pravděpodobnosti vzniku nových požadavků spolu s odhadem jejich velikosti. Touto problematikou se zabývají Gendreau et al. (1999), Gendreau a Potvin (1998), Savelsbergh a Sol (1998), Powell et al. (1995), Ichoua et al. (2000), Chen a Xu (2006) aj.

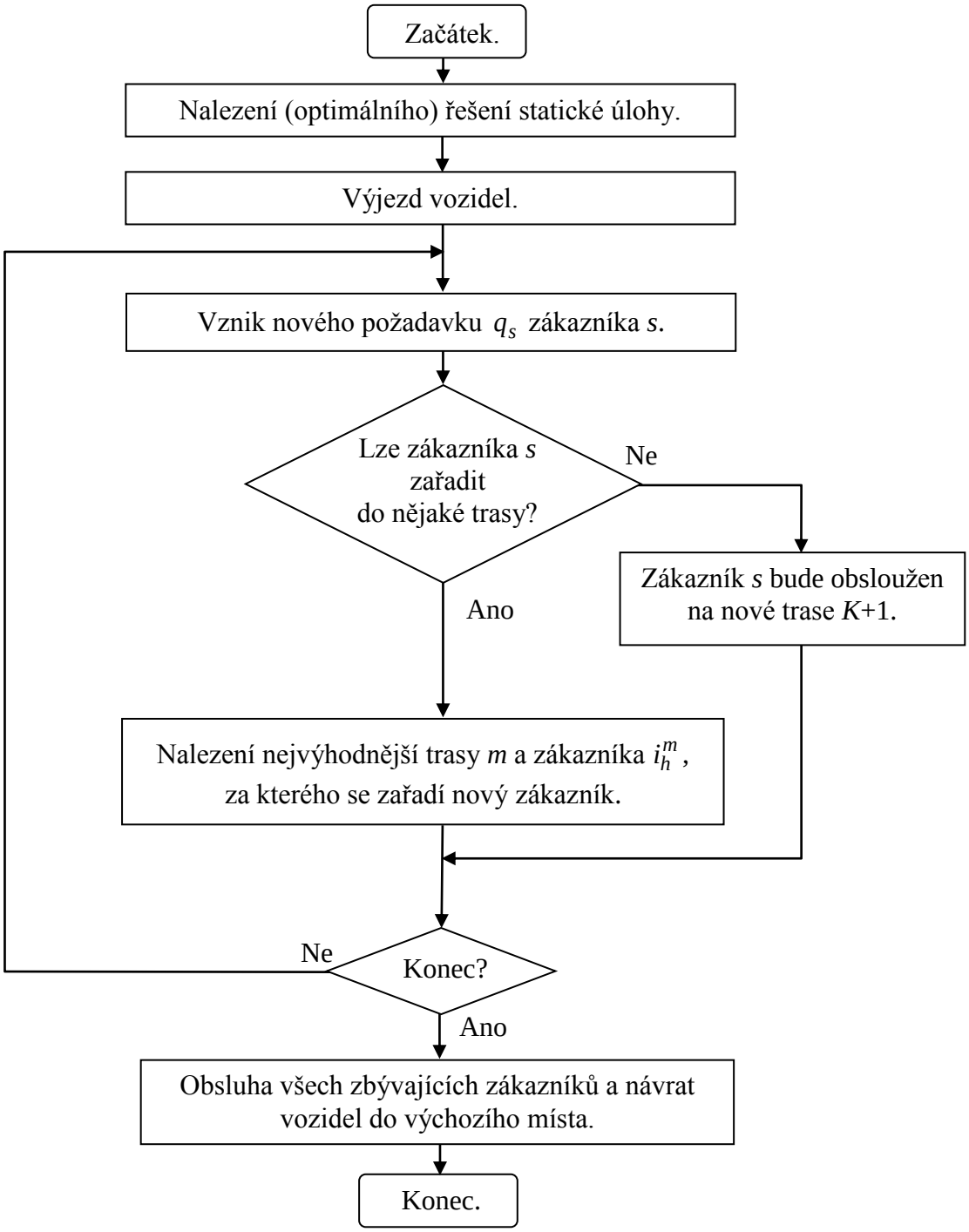
Pro dynamickou rozvozní úlohu s jedním vozidlem byl navržen re-optimalizační matematický model a vkládací metoda (Fábry, 2006a). V případě, že je ve výchozím místě k dispozici více vozidel, mohou na trasy vyjet současně. Jak bylo uvedeno výše, u rozvozních úloh, resp. úloh svozu, prakticky není možné následně vkládat do trasy místa s on-line požadavky z důvodu překročení kapacity vozidel. Výsledkem optimalizace statické úlohy je ve většině reálných aplikací totiž takové řešení, v němž jsou vozidla zcela či téměř zaplněna, a tudíž neexistuje prostor pro dodatečné zvyšování hodnoty nákladu vozidel. Jedinou možností je výše zmíněné vytváření rezerv, tj. snížení původních kapacit na základě odhadu celkového součtu nově vzniklých požadavků (např. na základě historických dat). V následujícím textu uvedeme princip vkládacího algoritmu pro dynamickou rozvozní úlohu s více vozidly v jednom výchozím místě.

Budeme předpokládat, že ve výchozím místě se nachází K vozidel s kapacitami V_k $(k = 1,2,...,K)$. V praxi se může jednat již o hodnoty uměle snížené o vytvořené rezervy. Po vyřešení statické úlohy vozidla začínají s realizací plánovaných tras. Posloupnost $U_k = \left\{ i_1^k, i_2^k, ..., i_{m_k}^k \right\}$ je tvořena místy, která má, v daný okamžik, vozidlo ještě navštívit na *k*-té trase $(k = 1,2,...,K)$. Při výjezdu vozidel zřejmě platí $i_1^k = i_{m_k}^k = 1\,(k = 1,2,...,K)$. V případě, že některé vozidlo zůstává ve výchozím místě (označme jej *l*), lze pro něj pro zjednodušení vytvořit fiktivní trasu, které odpovídá posloupnost $U_l = \{1,1\}$. Dále označme $L_k \leq V_k$ náklad, který bude mít na *k*-té trase vozidlo při návratu do výchozího místa. Při výjezdu vozidel jsou tyto hodnoty pro všechna vozidla nulové.

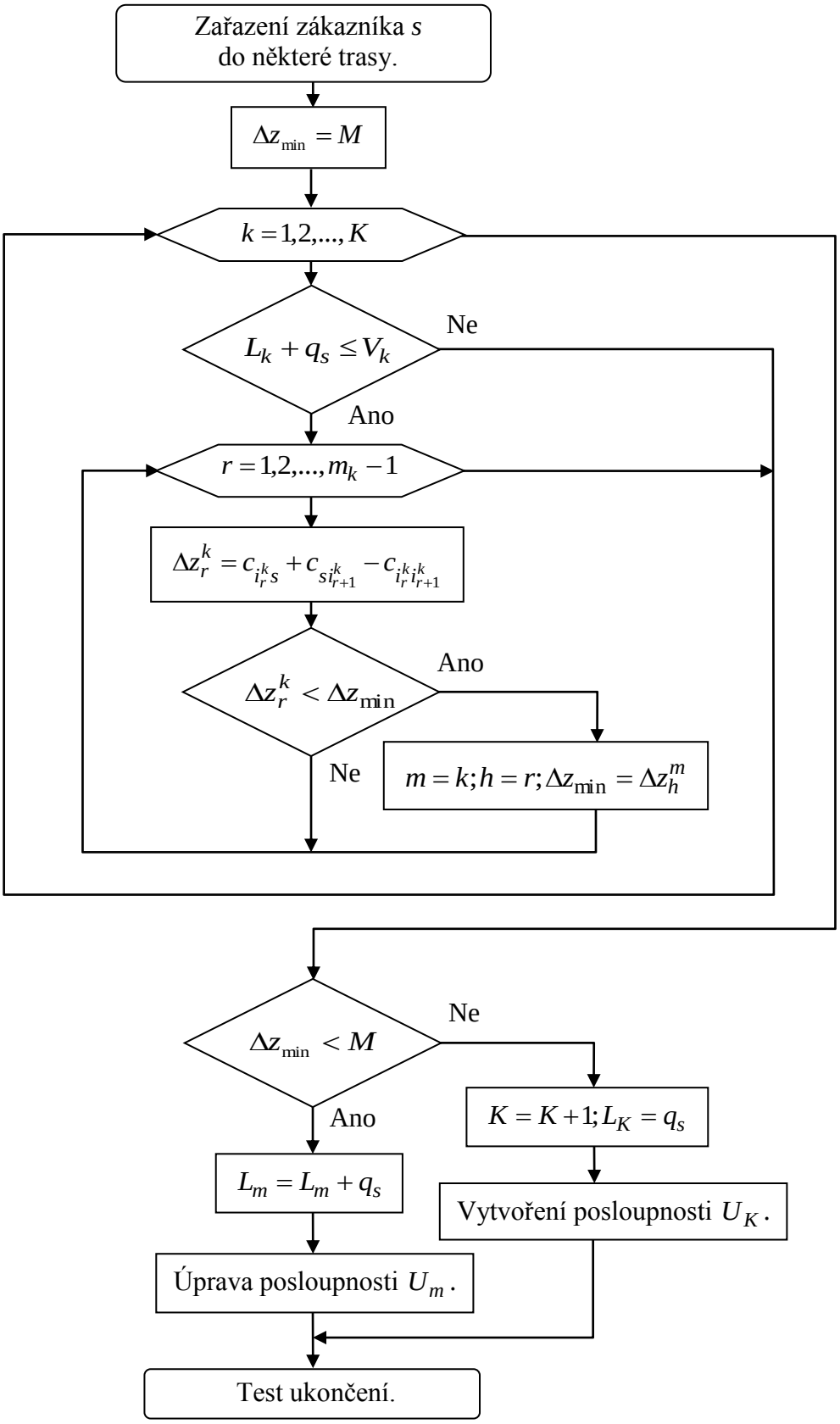
Po vzniku nového požadavku je nutné otestovat, zda je možné pro jeho splnění využít vozidlo v rámci některé z navržených tras, případně zahájit trasu dosud nevyužitého vozidla. Naložení požadavku *s*-tého zákazníka o velikosti q_s nesmí mít v žádném případě za následek překročení kapacity vozidla V_k na *k*-té trase $(k = 1,2,...,K)$. Cílem je nalézt nejvýhodnější trasu a zákazníka, za kterého bude vložen nový zákazník tak, aby prodloužení trasy bylo minimální. Pokud by kapacita vozidla byla překročena u všech tras, pak musí být nový zákazník obsloužen na nové trase, tedy po návratu některého z vozidel do výchozího místa[50].

Na obr. 3.1 je znázorněno hrubé schéma vkládací metody, algoritmus na obr. 3.2 je určen k výběru trasy pro vložení nového požadavku a místa na této trase, za které bude nový požadavek vložen. Pokud je požadavek vložen do fiktivní trasy, dojde k vytvoření reálné trasy. Parametr M v uvedeném schématu představuje vysokou konstantu. Celý proces končí v okamžiku, do kterého je možné přijímat další požadavky zákazníků, aby nedošlo k překročení pracovní doby distribuční firmy. V rámci metody je tedy nutné sledovat dobu, po kterou bude vozidlo ještě na trase před svým návratem do výchozího místa. Vozidla dokončí obsluhu všech zbývajících zákazníků a vracejí se do výchozího místa.

[50] V následujícím algoritmu uvažujeme jen situace, kdy není nutné dělit požadavek do více tras a lze jej uspokojit jedním vozidlem.



Obr. 3.1 – Vkládací algoritmus pro dynamickou rozvozní úlohu s více vozidly v jednom výchozím místě



Obr. 3.2 – Výběr trasy, na které bude obsloužen nový požadavek

Další dynamickou úlohou je problém, ve kterém je pro rozvoz či svoz určeno více výchozích míst, v nichž jsou umístěna vozidla o stejné či rozdílné kapacitě[51]. Úlohami tohoto typu se zabývají např. Laporte et al. (1988) a Savelsbergh a Sol (1998). Podobně jako v případě úlohy s více vozidly umístěnými v jednom výchozím místě lze i v úloze s několika výchozími místy minimalizovat celkovou ujetou vzdálenost nebo čas potřebný k obsluze všech zákazníků.

Na základě dříve uvedených matematických modelů a metod lze poměrně snadno odvodit modely pro řešení takové rozvozní úlohy; lze aplikovat i výše uvedený vkládací algoritmus. Při výjezdu vozidla z *k*-tého výchozího místa pro posloupnost $U_k = \left\{ i_1^k, i_2^k, ..., i_{m_k}^k \right\}$ platí $i_1^k = i_{m_k}^k = k \; (k = 1,2,...,K)$. Pro vozidlo, které zůstává ve svém výchozím místě (označme jej *l*), je vytvořena fiktivní trasa a definována posloupnost $U_l = \left\{ i_1^l, i_2^l \right\}$, kde $i_1^l = i_2^l = l$. Tyto posloupnosti se pak dynamicky upravují podle toho, jak jsou zařazovány nově příchozí požadavky do realizovaných tras. Schéma algoritmu je v podstatě shodné s výše uvedeným postupem.

Již po optimalizaci statické úlohy mohou být vozidla na všech trasách prakticky zcela zaplněna a zařazení dalších zákazníků je zcela vyloučeno. Jejich obsluhu je pak nutné zajistit v nově vytvořených trasách. Tato strategie je nevýhodná především v situacích, kdy vozidlo jede plně naloženo kolem zákazníka velmi vzdáleného od výchozího místa a musí se k němu vrátit na některé z příštích tras, případně jej musí obsloužit jiné vozidlo, pro které to ovšem může být také značně nevýhodné. Pokud jsou v takovém případě některá vozidla zaplněna více a jiná méně, může být výhodné, ovšem pouze pro úlohy menšího rozsahu, použití reoptimalizačního modelu (Fábry, 2006b). Předpokládané požadavky jsou přesunuty na jiná vozidla a výše zmíněný vzdálený požadavek může být obsloužen vozidlem, které se pohybuje v jeho blízkosti. Velmi efektivním nástrojem je v tomto případě také již dříve zmíněná metoda výměn.

Jak bylo uvedeno výše, dynamické rozvozní úlohy jsou velice komplikované díky neznalosti velikosti požadavků, které vznikají až v průběhu realizace naplánovaných tras. Strategie vytváření rezerv v kapacitě vozidla pro podobné případy může být z dlouhodobějšího hlediska (např. vzhledem k pracovní době firmy) výhodná ve smyslu minimalizace celkové vzdálenosti i při větším počtu tras. V popsaných modelech lze provést jednoduchou změnu tak, že se po-

[51] V každém výchozím místě je právě jedno vozidlo.

stupně zvyšují hodnoty parametrů V_k, představující kapacity vozidel, např. z poloviční hodnoty až na skutečnou hodnotu. Tyto úvahy, které jsou podmíněny charakterem reálné firmy a jejích zákazníků, si jistě zaslouží důkladné rozpracování v podobě vyhodnocení simulačních experimentů. To samozřejmě předpokládá znalost pravděpodobnostních rozdělení náhodných veličin, týkajících se vzniku nových požadavků.

3.5 Rozvozní úloha s dělenou dodávkou

Ve všech výše uvedených rozvozních úlohách byl zaveden předpoklad, že vyřízení požadavku určitého zákazníka nesmí být rozloženo na více částí, tj. do více tras. Tento předpoklad, který vyžaduje, aby v matematickém modelu byly formulovány podmínky zakazující navštívit některého zákazníka více než jednou, je nutné opustit v případě, že požadavek některého zákazníka převyšuje kapacitu vozidla. Ovšem i v situacích, kdy požadavek není nutné dělit, neboť jej lze uspokojit v rámci jediné trasy, existují výhody dělení dodávky. Především v dynamických úlohách hraje možnost rozdělení požadavku do několika tras významnou roli a může dojít k výraznému snížení přepravních nákladů, resp. zkrácení celkové vzdálenosti ujeté všemi vozidly. Statickou verzi tohoto problému lze v literatuře nalézt pod názvem rozvozní úloha s dělenou dodávkou (SDVRP)[52].

Pro řešení této úlohy byl navržen algoritmus tabu search (Archetti et al, 2006). Otázkou řešitelnosti a rozložitelnosti úlohy s dělenou dodávkou se zabývali Archetti et al. (2005, [125]). Analýzu úspor vzniklých díky možnosti rozdělení dodávky mezi více vozidel provedli Dror a Trudeau (1989), kteří zároveň prezentovali heuristický algoritmus pro řešení úloh tohoto typu.

3.5.1 Exaktní přístup

Nechť existuje jediné výchozí místo, v němž je umístěno několik vozidel určených ke svozu zboží. Cílem je minimalizovat celkovou ujetou vzdálenost. Je poměrně snadné tuto úlohu rozšířit i na jiné typy úloh týkajících se minimalizace času potřebného k obsluze všech zákazníků, případně na úlohy, v nichž existuje více výchozích míst a/nebo je použito více vozidel s rozdílnou kapacitou. Pro řešení statické úlohy s K vozidly o stejné kapacitě V umístěnými v jednom výchozím místě, která jsou určena pro svoz zboží od n-1 zákazníků, lze použít následující matematický model (Fábry, 2005a):

[52] Split Delivery Vehicle Routing Problem.

minimalizovat $z = \sum_{k=1}^{K} \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}^{k},$ (3.30)

za podmínek

$$\sum_{j=2}^{n} x_{1j}^{k} \leq 1, \quad k = 1,2,...K, \tag{3.31}$$

$$\sum_{i=1}^{n} x_{ij}^{k} = \sum_{i=1}^{n} x_{ji}^{k}, \quad j = 2,3,...,n, \quad k = 1,2,...K, \tag{3.32}$$

$$u_i^k + Q_j^k - V(1 - x_{ij}^k) \leq u_j^k, \quad i = 1,2,...,n, \quad j = 2,3,...,n, \quad i \neq j, \quad k = 1,2,...,K, \tag{3.33}$$

$$u_1^k = 0, \quad k = 1,2,...,K, \tag{3.34}$$

$$0 \leq Q_i^k \leq u_i^k \leq V, \quad i = 2,3,...,n, \quad k = 1,2,...,K, \tag{3.35}$$

$$\sum_{k=1}^{K} Q_i^k = q_i, \quad i = 2,3,...,n, \tag{3.36}$$

$$0 \leq Q_i^k \leq q_i \sum_{j=1}^{n} x_{ij}^k, \quad i = 2,3,...,n, \quad k = 1,2,...K, \tag{3.37}$$

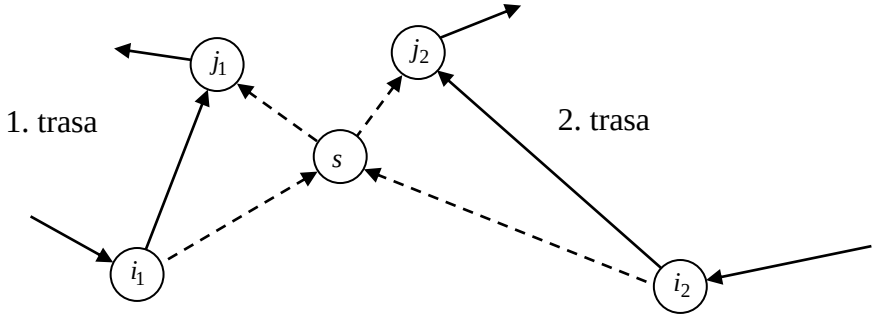
$$x_{ii}^k = 0, \quad i = 1,2,...,n, \quad k = 1,2,...,K, \tag{3.38}$$

$$x_{ij}^k \in \{0,1\}, \quad i, j = 1,2,...,n, \quad k = 1,2,...,K. \tag{3.39}$$

Účelová funkce (3.30) a omezující podmínky (3.31) a (3.32) mají stejný význam jako podmínky ve výše uvedených modelech. Proměnná Q_i^k v modelu představuje část požadavku q_i *i*-tého zákazníka, která bude odvezena vozidlem na trase *k*. Podmínky (3.36) zajišťují, aby všechna vozidla tímto způsobem odvezla od každého zákazníka požadované množství.

Nerovnosti (3.37) mají dvojí význam. V případě, že *k*-té vozidlo neobsluhuje *i*-tého zákazníka, tj. $x_{ij}^k = 0$ pro $j = 1,2,...,n,$ pak $Q_i^k = 0.$ Pokud se na *k*-té trase od *i*-tého zákazníka odváží část požadavku $Q_i^k > 0,$ pak musí existovat alespoň jedno místo, do kterého vozidlo pojede v rámci této trasy přímo od *i*-tého zákazníka, tj. $x_{ij}^k = 1$ pro některý index *j*. Na rozdíl od klasické úlohy VRP (s nedělenou dodávkou), v níž vystupují proměnné u_i, musí být v tomto modelu zavedeny proměnné u_i^k, neboť *i*-té místo může být navštíveno několikrát, a tudíž pro každé vozidlo *k*, které z tohoto místa něco odváží, musí být použita jiná proměnná. Těmto proměnným také odpovídají podmínky (3.33) – (3.35). Jejich význam je opět zřejmý z výše uvedených matematických modelů pro VRP.

Jak již bylo uvedeno, možnost rozdělení požadavku zákazníků do několika tras je výhodná zejména u dynamických rozvozních úloh. Pro zařazení nově vzniklého požadavku do tras vozidel může být dokonce efektivní použít heuristický vkládací algoritmus, jehož aplikace je v případě standardních dynamických rozvozních úloh prakticky vyloučena. Na obr. 3.3 jsou znázorněny trasy dvou vozidel. První trasa vede přes uzly i_1 a j_1, druhá trasa pak přes uzly i_2 a j_2.



Obr. 3.3 – Zařazení nově vzniklého požadavku do naplánovaných tras

Uzel *s* představuje nově vzniklý požadavek. Předpokládejme, že ani jedno vozidlo nemůže tento požadavek uspokojit, tj. zařadit zákazníka do své trasy, z důvodu překročení kapacity. Z hlediska standardní rozvozní úlohy by tedy bylo nutné nového zákazníka obsloužit na jiné trase, což ovšem může být zcela nevýhodné. Lze samozřejmě použít re-optimalizační algoritmus, který obě trasy (kromě dalších tras) upraví tak, že nově vzniklý požadavek bude efektivně uspokojen. V případě rozsáhlých úloh ovšem optimalizační postupy NP-obtížných úloh naráží na problém s řešitelností (viz podkapitola 2.7). Pokud by bylo možné nový požadavek rozdělit tak, že se obě menší části podaří naložit na obě vozidla bez překročení jejich kapacity (myšleno i v další časti trasy), přínos této možnosti je evidentní. Rozšíření re-optimalizačního modelu je uvedeno v práci Fábry (2006a).

3.5.2 Heuristická metoda pro úlohu s dělenou dodávkou

Stejně jako v jiných okružních a rozvozních úlohách, také v reálných úlohách s dělenou dodávkou není často možné nalézt optimální řešení v přijatelném čase. V takovém případě se opět nabízí použití některé heuristické metody. Dror a Trudeau (1989) uvádějí následující postup.

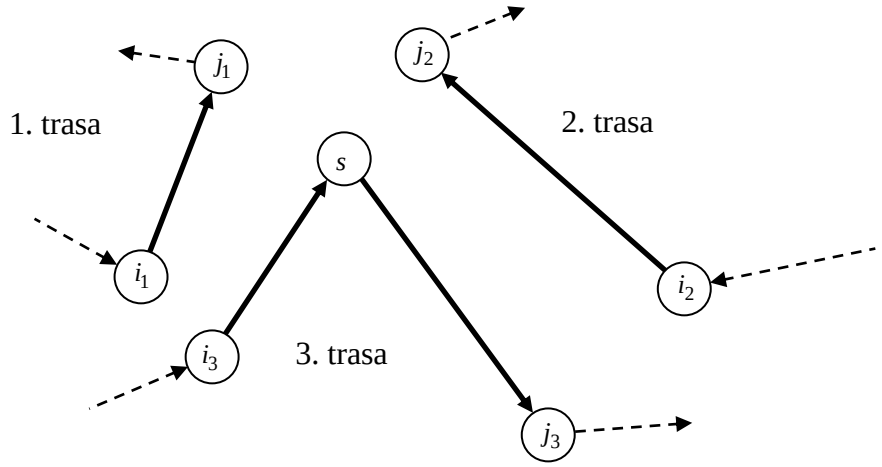
Na začátku je nutné získat přípustné řešení dané rozvozní úlohy bez možnosti dělení dodávky. Může se jednat o trasy nalezené některou ze známých heuristických metod či dokonce

o trasy optimální získané řešením modelu (3.1) – (3.8)[53]. Předpokládejme existenci tří tras; vozidlo na první trase má volnou kapacitu V_1, druhé vozidlo volnou kapacitu V_2. Nechť na třetí trase existuje zákazník *s*, pro jehož dodávku q_s, realizovanou na této trase[54], platí:

$$q_s \leq V_1 + V_2, \tag{3.40}$$

tj. tuto dodávku lze přeložit na vozidla jedoucí v první a druhé trase.

Dále označme ve výchozím řešení i_1 a j_1 po sobě jdoucí uzly na trase 1, i_2 a j_2 po sobě jdoucí uzly na trase 2, i_3 uzel na trase 3, který bezprostředně předchází uzlu *s* a konečně j_3 uzel, který bezprostředně následuje po uzlu *s* (obr. 3.4).



Obr. 3.4 – Výchozí přípustné řešení

Součet vzdáleností, týkající se zmíněných uzlů na všech třech trasách lze vyjádřit takto:

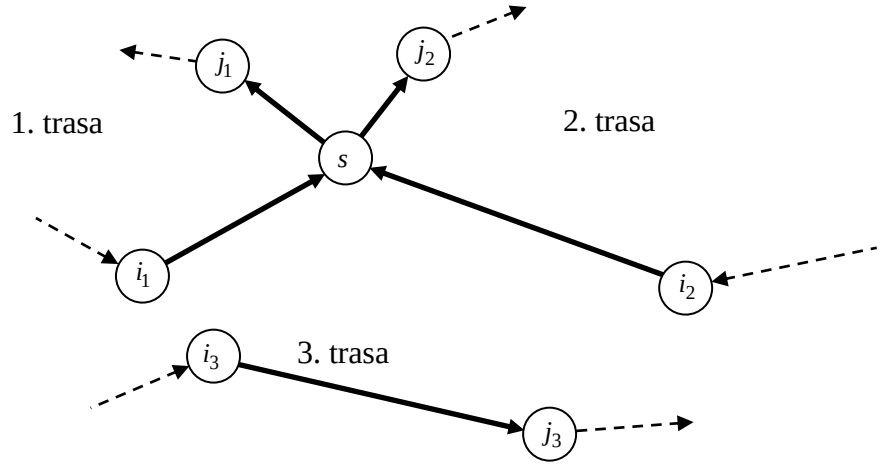
$$z_1 = c_{i_1 j_1} + c_{i_2 j_2} + c_{i_3 s} + c_{s j_3}. \tag{3.41}$$

Jestliže při respektování omezení (3.40) rozdělíme původní dodávku q_s do uzlu *s* na trase 3 na vozidla jedoucí po trasách 1 a 2, získáme nový součet vzdáleností:

$$z_2 = c_{i_1 s} + c_{s j_1} + c_{i_2 s} + c_{s j_2} + c_{i_3 j_3}. \tag{3.42}$$

[53] Uvedený postup lze samozřejmě modifikovat i pro úlohy s různými kapacitami vozidel a úlohy s několika výchozími místy.
[54] V průběhu aplikace algoritmu se může jednat o část celkové dodávky místa *s*, protože další části mohou být realizovány na zbývajících trasách.

Hodnota $\Delta z = (z_1 - z_2) > 0$ je pak úspora vzdálenosti, které bylo dosaženo tím, že původní dodávka do uzlu *s* byla přeložena na vozidla na trasách 1 a 2, čímž byl uzel *s* z trasy 3 zcela vyloučen (obr. 3.5). Tento postup lze aplikovat do té doby, dokud je možné zlepšovat řešení z hlediska celkové délky všech tras.



Obr. 3.5 – Nové řešení s dělenou dodávkou

4 Přepravní problémy

Speciální skupinu tvoří úlohy, v nichž je nutné přepravit materiál, výrobky či zboží z jednoho či několika míst do jedné či několika destinací. V některých případech není dokonce nutné ani vytvářet okruhy. Přestože většinou existují výchozí místa jako stanoviště pro vozidla, samotné převážení nákladu se děje nezávisle na těchto depotech v tom smyslu, že cílem není odvézt nějaký náklad z výchozího místa či do něj naopak nějaký náklad svézt, ale místa nakládky a vykládky jsou umístěna na trase. Navíc se může velice snadno stát, že jedno místo může být jak místem naložení, tak zároveň místem vykládky; tato situace má samozřejmě svůj význam jen v případě, že se jedná o přepravu nehomogenního zboží.

Do skupiny úloh, která se v anglické literatuře obecně označuje pojmem Pickup and Delivery Problem (PDP), lze zahrnout obrovské množství problémů, které mají jedno společné, a sice skutečnost, že požadavek je zadán místem vyzvednutí (pickup) a místem doručení zásilky (delivery) či vyložení osob (drop off). Právě tím se tyto úlohy zásadně odlišují od rozvozních úloh, které byly obsahem předchozí kapitoly, a v nichž se jedná o rozvoz nebo svoz jednotek z nebo do výchozího místa a každý požadavek je tedy určen pouze jedním místem, kam má vozidlo přijet.

Cordeau et al. (2008) vyjadřují základní klasifikaci úloh PDP. První třída úloh odpovídá slovnímu spojení „one-to-one“, které je typické pro úlohy kurýrní služby (viz podkapitola 4.2). Požadavek je zadaný dvěma místy, a sice místem vyzvednutí a místem doručení. Předpokládáme, že v žádném z těchto dvou míst nelze vyzvednout ani doručit žádné jiné zásilky, týkající se jiných míst. Pokud se jedná o převoz osob, např. hendikepovaných, pak se tato úloha označuje jako Dial-a-Ride Problem.

„Many-to-many“ PDP je úlohou, která se podobá dále uvedenému Transhipment problému. Jsou zadána místa, v nichž je potřeba naložit určitá množství homogenního produktu, a jiná místa, kam je nutné určitá množství tohoto produktu doručit. Na rozdíl od Transhipment problému není zadána kapacita hran, ale je zadána kapacita vozidla. Navíc existuje depot, odkud toto vozidlo na trasu vyjíždí (prázdné, neboť ve výchozím místě není zásoba produktu), a do kterého se vrací, tj. jedná se o okružní typ úlohy. Opět se místa rozdělují na množinu těch, kde se produkt pouze nakládá, a množinu těch, kde se produkt pouze vykládá.

Ještě se můžeme zmínit o úloze PDP typu „one-to-many-to-one“. V každém místě lze (obecně) určitou zásilku či zásilky naložit a jinou zásilku či zásilky vyložit[55]. Opět je zadáno výchozí místo, odkud prázdné vozidlo vyjíždí a kam se později také prázdné vrací. V podkapitole 4.3 je řešena úloha, v níž je právě tento předpoklad nutné opustit.

Nejprve se ale podívejme na speciální přepravní úlohu, označovanou jako Transhipment Problem.

4.1 Transhipment Problem

Tato přepravní úloha, jejíž základní verzi popisují např. Eiselt a Sandblom (2000), patří mezi tzv. tokové úlohy a není tedy typickým představitelem rozvozních úloh. Cílem je přepravit homogenní produkt od dodavatelů k odběratelům prostřednictvím distribuční sítě s omezenou kapacitou. Jako kritérium je zvolena funkce celkových přepravních nákladů. Distribuční síť je vyjádřena grafem $G = \{U, H\}$, kde U je množina uzlů a H množina hran. Každému uzlu $i \in U$ je přiřazena hodnota q_i, uzly jsou rozděleny do tří množin:

1) $i \in U^+$, pokud uzel i je dodavatelem produktu; hodnota $q_i > 0$ je jeho kapacita,

2) $i \in U^-$, pokud uzel i je odběratelem produktu; hodnota $q_i < 0$ určuje jeho požadavek,

3) $i \in U^0$, pokud uzel i je tranzitní; hodnota $q_i = 0$.

Evidentně platí: $U = U^+ \cup U^- \cup U^0$ a $U^+ \cap U^- \cap U^0 = \{\ \}$. Každé hraně $h_{ij} \in H$ jsou přiřazeny náklady c_{ij} na přepravu jedné jednotky produktu po této hraně a dále maximální propustnost (kapacita) hrany κ_{ij}. Pro následující model předpokládejme neorientovaný graf:

$$\text{minimalizovat} \quad \sum_{\substack{i \in U \\ h_{ij} \in H}} \sum_{j \in U} c_{ij} x_{ij}, \tag{4.1}$$

za podmínek

$$\sum_{\substack{j \in U \\ h_{ij} \in H}} x_{ij} \; - \sum_{\substack{j \in U \\ h_{ji} \in H}} x_{ji} = q_i \ , \quad i \in U, \tag{4.2}$$

$$0 \le x_{ij} \le \kappa_{ij}, \quad h_{ij} \in H. \tag{4.3}$$

Hodnota proměnné x_{ij} určuje tok hranou h_{ij}, tedy počet jednotek produktu přepravovaného mezi místy i a j. Všechny proměnné musí respektovat zadanou kapacitu (4.3). Minimalizační

[55] Všechny zásilky, které se na trase postupně naloží, se musí na trase vyložit.

účelová funkce (4.1) představuje celkové přepravní náklady. Rovnice (4.2) jsou bilančními podmínkami pro každý uzel. Pro dodavatele musí být počet jednotek, které jsou z něj přepravovány, o q_i vyšší než počet jednotek do něj přepravených. Pro odběratele je tomu naopak (na pravé straně rovnic jsou záporná čísla). Pro tranzitní uzly platí, že bilance dovozu a odvozu musí být vyrovnaná. Je zřejmé, že nutnou podmínkou přípustnosti řešení této úlohy je následující rovnice:

$$\sum_{i \in U^+} q_i + \sum_{i \in U^-} q_i = 0, \tag{4.4}$$

tedy součet všech kapacit dodavatelů je roven součtu všech požadavků odběratelů, neboli celková nabídka je rovna celkové poptávce. Pokud by rovnice (4.4) nebyla splněna, je uvedený model možné použít pouze v případě, že přidáme fiktivní uzel. Dalšími podmínkami, které musí být splněny, jsou nerovnice:

$$\sum_{\substack{j \in U \\ h_{ij} \in H}} \kappa_{ij} \geq |q_i|, \quad i \in U, \tag{4.5}$$

tj. součet kapacit hran incidentních s daným uzlem musí být minimálně takový, jako je jeho kapacita, resp. požadavek (pro tranzitní uzly je tato podmínka splněna vždy).

Zajímavým rozšířením je kontejnerová verze výše uvedeného problému. Předpokládejme, že produkt je přepravovaný v kontejnerech (vozidlech, vagónech) o kapacitě *V*. Přepravní náklady c_{ij} jsou vztaženy na přepravu jednoho kontejneru[56] po hraně h_{ij}. V takové úloze je nutné zavést pro každou hranu celočíselnou proměnnou, jejíž hodnota udává počet kontejnerů použitých pro přepravu produktu po této hraně. Účelová funkce bude upravena následovně:

$$\text{minimalizovat} \quad \sum_{\substack{i \in U \\ h_{ij} \in H}} \sum_{j \in U} c_{ij} y_{ij}. \tag{4.1a}$$

Navíc musí být k výše uvedenému modelu přidána soustava omezujících podmínek, které bilancují využití kontejnerů pro přepravu množství produktu a podmínky celočíselnosti pro proměnné y_{ij}:

$$x_{ij} \leq V y_{ij}, \quad h_{ij} \in H, \tag{4.6}$$

$$y_{ij} \in Z_0^+, \quad h_{ij} \in H. \tag{4.7}$$

[56] V případě vozidla se jedná o náklady na jízdu vozidla po hraně, v případě vagónu se jedná např. o jeho pronájem.

V reálných úlohách většinou kapacity hran představují horní meze počtu kontejnerů (např. vagónů, které lze na daném železničním úseku použít), namísto počtu jednotek, který je předpokládán v uvedeném modelu. V takovém případě je zapotřebí vhodným způsobem upravit podmínky (4.3) a (4.7).

V každém případě model výše uvedeného kontejnerového přepravního problému neřeší logistickou otázku vozového parku (obecně počtu kontejnerů), které jsou v jednotlivých místech k dispozici, ani to, jak se do těchto míst dostanou. V případě reálné úlohy, v níž se tento model řeší opakovaně (např. každý den) s různými kapacitami dodavatelů a požadavky odběratelů, je nutné určit pro každý uzel další údaj, a sice počet kontejnerů, které budou daný den v daném místě k dispozici. Následující části kapitoly jsou zaměřeny na přepravní úlohy s vyzvednutím a doručením, jejichž řešení (na rozdíl od řešení úlohy Transhipment Problem) spočívá v nalezení okruhů.

4.2 Úloha kurýrní služby

Podstata tohoto problému odpovídá do jisté míry úloze Dial a Ride Problem (DARP), v níž jsou ovšem místo věcných zásilek přepravovány osoby (tzv. přeprava od dveří ke dveřím). V takovém případě je vždy zapotřebí uvažovat kapacitu vozidla a většinou také časová okna pro nástup (pickup) a výstup (drop off) každého klienta. Přesný algoritmus pro statickou úlohu, v níž jsou všechny požadavky na přepravu osob známé předem, uvedl Psaraftis (1983). Statickou úlohou, v níž je pro přepravu osob k dispozici několik vozidel umístěných v jednom výchozím místě se zabýval Cordeau (2006), který pro řešení této úlohy popsal metodu větvení a řezů. Pro úlohu s více vozidly a časovými okny navrhli Toth a Vigo (1997) paralelní heuristický vkládací algoritmus, Jorgensen et al. (2007) použili pro řešení genetické algoritmy. Možnost využití constraint programming (např. Fábry a Pelikán, 2003) při řešení DARP uvádí Berbeglia et al. (2011). Přestože je v povědomí lidí kurýrní služba většinou spojena s přepravou věcných zásilek, budeme, z důvodu neexistence přesného českého ekvivalentu výše uvedeného anglického označení úloh DARP, zmíněné úlohy zahrnovat pod všezahrnující název úlohy kurýrní služby (Fábry, 2006a). V následujícím textu se soustředíme na nekapacitní a kapacitní úlohy, včetně jejich dynamického rozšíření.

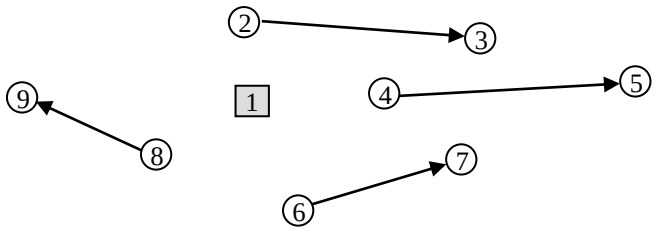
4.2.1 Nekapacitní úloha kurýrní služby s jedním vozidlem

Jestliže kapacita vozidla významně převyšuje velikost předpokládaného nákladu, lze při formulaci matematického modelu vycházet z definice úlohy obchodního cestujícího. V případě

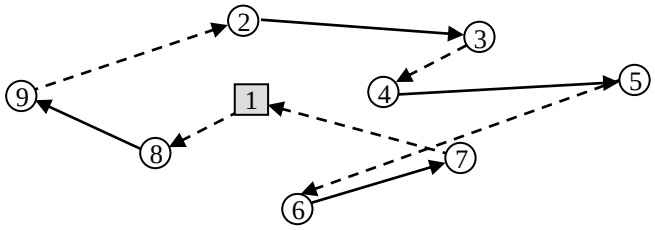
kurýrní služby je ovšem nutné respektovat správné pořadí navštívených míst, v nichž se zásilky vyzvedají a doručují. Uvažujeme-li statickou verzi úlohy, jsou všechny požadavky na přepravu známé předem, tj. před zahájením optimalizace. V průběhu realizace nalezené trasy do ní není možné zařazovat další požadavky.

4.2.1.1 Statická úloha

Na obr. 4.1 je ukázka úlohy, v níž se mají doručit čtyři zásilky. Důležité je očíslování uzlů, které bude dodrženo ve všech modelech. Uzly se sudými čísly odpovídají místům, v nichž se má zásilka vyzvednout, uzly s lichými čísly označují místa, kam se má zásilka doručit. Tato čísla jsou vždy o 1 vyšší, než má uzel, v němž se vyzvedává příslušná zásilka[57]. Určení zásilky je v grafu naznačeno orientovanou hranou. Uzel č. 1 je depotem, v němž je připraveno jedno vozidlo pro vyzvednutí a doručení všech zásilek. Protože se jedná o úlohu přepravní typu „one-to-one“, předpokládáme, že všechny uzly jsou různé. Zásilky nemusí být samozřejmě doručeny bezprostředně po jejich vyzvednutí (tato situace je znázorněna na obr. 4.2).



Obr. 4.1 – Požadavky zákazníků na doručení zásilek



Obr. 4.2 – Doručení zásilek bezprostředně po jejich vyzvednutí

Následující model (Fábry, 2010) vychází z označení odpovídajícího obr. 4.1, tedy zásilka je definována dvojicí uzlů i a $i+1$, kde $i>1$, sudé. Předokládejme, že známe n míst vyzvednutí a n míst doručení zásilek, celkem tedy $2n + 1$ míst (včetně depotu). Dále známe

[57] Existují samozřejmě alternativní možnosti jak označit zmíněné dvojice míst. Cordeau (2006) zavádí toto označení: $P = \{1,2,\ldots,n\}$ je množina míst, v nichž se zásilky vyzvednou, $D = \{n+1,n+2,\ldots,2n\}$ množina míst, kam se doručí, n je počet zásilek. Indexem 0 je označeno výchozí místo. Každá zásilka je tedy v grafu určena dvojicí uzlů i a $n+i$, mezi nimiž vede orientovaná hrana.

nejkratší vzdálenosti c_{ij} mezi místy *i* a *j*. Protože se jedná o využití jediného vozidla, jehož kapacitu nemusíme, vzhledem k nulovým velikostem požadavků, uvažovat, jedná se o modifikaci modelu TSP:

$$\text{minimalizovat} \quad z = \sum_{i=1}^{2n+1} \sum_{j=1}^{2n+1} c_{ij} x_{ij}, \tag{4.8}$$

za podmínek

$$\sum_{j=1}^{2n+1} x_{ij} = 1, \quad i = 1,2,...,2n+1, \tag{4.9}$$

$$\sum_{i=1}^{2n+1} x_{ij} = 1, \quad j = 1,2,...,2n+1, \tag{4.10}$$

$$u_i - u_j + (2n+1)x_{ij} \le 2n, \quad i = 1,2,...,2n+1, \quad j = 2,3,...,2n+1, \tag{4.11}$$

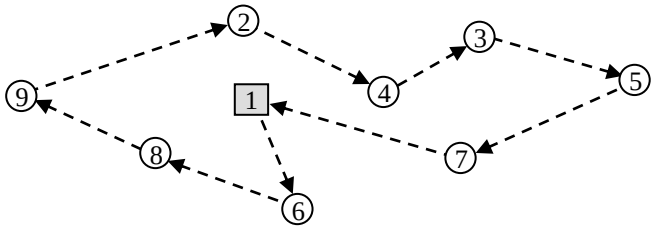
$$u_{2i} \le u_{2i+1}, \quad i = 1,2,...,n, \tag{4.12}$$

$$u_1 = 0, \tag{4.13}$$

$$x_{ij} \in \{0,1\}, \quad i, j = 1,2,...,2n+1, \tag{4.14}$$

$$u_i \in R_0^+, \quad i = 1,2,...,2n+1. \tag{4.15}$$

Proměnná x_{ij} je bivalentní proměnná nabývající hodnoty 1 v případě, že vozidlo jede do místa *j* z místa *i*, hodnoty 0 v opačném případě. Model obsahuje účelovou funkci a podmínky, které jsou obdobou modelu (2.1) – (2.6). Jediný rozdíl spočívá v přidání nerovností (4.12), které zaručují splnění výše uvedeného požadavku, že zásilky mohou být doručeny teprve až po jejich vyzvednutí. Pokud uvažujeme úlohu znázorněnou na předchozích obrázcích, můžeme získat jako optimální řešení[58] okruh uvedený na obr. 4.3.



Obr. 4.3 – Optimální trasa pro doručení zásilek

V této konkrétní ukázce bude zásilka z uzlu č. 6 vyzvednuta jako první a doručena do uzlu č. 7 jako poslední.

[58] Jedná se samozřejmě pouze o demonstrační příklad, který předpokládá euklidovské nejkratší vzdálenosti mezi uzly.

V reálných úlohách jsou většinou uplatňována časová okna. Nejprve uvažujme obecnější úlohu, v níž je pro každou zásilku zadaný interval, během kterého je nutné ji vyzvednout, a interval, v němž je nutné ji doručit. Kromě vzdáleností c_{ij} je nutné znát i přejezdové doby t_{ij} mezi všemi místy. Protože i v tomto případě je model rozšířením úlohy TSPTW (viz podkapitola 2.3), uvedeme jen omezení, které je nutné přidat pro zajištění návaznosti vyzvednutí a následného doručení každé zásilky:

$$\tau_{2i} \leq \tau_{2i+1}, \quad i = 1,2,...,n. \tag{4.16}$$

Podobně jako v úloze TSPTW, proměnná τ_i udává okamžik, v němž vozidlo navštíví místo *i*. Uvedené podmínky zaručí, že každá zásilka bude doručena až po jejím vyzvednutí. I v tomto případě může vozidlo navštívit mezi vyzvednutím a doručením konkrétní zásilky několik jiných míst, což podmínky (4.16) umožňují. Podmínky týkající se časových oken mohou být v reálných úlohách zadány i jiným způsobem. Jednou z možností je definovat nejdříve možný termín pro vyzvednutí zásilky a nejpozději přípustný termín pro její doručení. Standardní „hard" podmínky pro splnění časových oken pak budou přepsány do následujících nerovností:

$$e_{2i} \leq \tau_{2i}, \quad i = 1,2,...,n, \tag{4.17}$$

$$\tau_{2i+1} \leq l_{2i+1}, \quad i = 1,2,...,n, \tag{4.18}$$

tj. není zadána horní mez pro okamžik vyzvednutí zásilky a rovněž neexistuje dolní mez pro okamžik jejího doručení. Také v případě kurýrní služby je možné v účelové funkci použít penalizaci za nedodržení odpovídajících termínů a uvažovat tzv. „soft" časová okna.

V reálných úlohách je použití výše uvedeného matematického modelu komplikované, a to vzhledem k časové náročnosti řešení NP-obtížných úloh, mezi něž úloha kurýrní služby, stejně jako TSP, patří. Pokud není možné najít optimální řešení v reálném čase, je nutné se spokojit s řešením po předčasném ukončení optimalizačního výpočtu nebo použít některý heuristický algoritmus. Budou uvedeny modifikace dvou heuristických algoritmů pro řešení úlohy TSP, které byly předmětem podkapitoly 2.7.

Modifikovaná metoda nejbližšího souseda

Nechť U^* je posloupnost míst, odpovídající vygenerované trase, dále 1 je výchozí uzel, *j* je poslední uzel na vytvářené trase a *z* je celková délka trasy. Množina U_N bude obsahovat dosud nezařazené uzly, které je možné v daném kroku zařadit, tj. na začátku v ní budou jen sudá čísla, která představují místa vyzvednutí zásilek.

Krok 1: $U^* = \{1\}$; $U_N = \{2,4,...,2n\}$; $j = 1$; $z = 0$.

Krok 2: Najdi místo *s* takové, že $c_{js} = \min_{r \in U_N} c_{jr}$.

$U^* = U^* + \{s\}$; $U_N = U_N - \{s\}$; $z = z + c_{js}$.

Když *s* je sudé, pak $U_N = U_N + \{s+1\}$.

$j = s$.

Krok 3: Když je $U_N = \emptyset$, pak jdi na krok 4, jinak jdi na krok 2.

Krok 4: $U^* = U^* + \{1\}$; $z = z + c_{j1}$. Konec.

Modifikovaná vkládací metoda

Nechť $U^* \subseteq U$ je posloupnost míst (uzlů), které jsou zařazeny do trasy, u_i je prvek posloupnosti U^* na *i*-té pozici, množina U_N bude obsahovat dosud nezařazené uzly, dále 1 je výchozí uzel a *z* je celková délka trasy.

Krok 1: $U_N = \{2,3,...,2n+1\}$

Najdi libovolný uzel $s \neq 1$, např. takový, pro který platí: $c_{1s} = \max_{r \neq 1} c_{1r}$.

$U^* = \{1, s, 1\}$; $z = c_{1s} + c_{s1}$; $U_N = U_N - \{s\}$.

Krok 2: $\Delta z = \infty$.

Pro každé $r \in U_N$ opakuj:

a) pokud je *r* sudé a v posloupnosti U^* existuje uzel *r* + 1 na *s*-té pozici, pak
$\Delta z_{rj} = \min_{i=1,2,...,s-1} (c_{u_i,r} + c_{r,u_{i+1}} - c_{u_i,u_{i+1}})$,

b) pokud je *r* liché a v posloupnosti U^* existuje uzel *r* – 1na *s*-té pozici, pak
$\Delta z_{rj} = \min_{i=s,s+1,...,|U^*|-1} (c_{u_i,r} + c_{r,u_{i+1}} - c_{u_i,u_{i+1}})$,

c) jinak
$\Delta z_{rj} = \min_{i=1,2,...,|U^*|-1} (c_{u_i,r} + c_{r,u_{i+1}} - c_{u_i,u_{i+1}})$.

Pokud $\Delta z_{rj} < \Delta z$, pak

$\Delta z = \Delta z_{rj}$; $v = r$; $w = j$.

Do posloupnosti U^* přidej uzel *v* za prvek u_w.

$U_N = U_N - \{v\}$; $z = z + \Delta z$.

Krok 3: Když je $U_N = \emptyset$, pak jdi na krok 4, jinak jdi na krok 2.

Krok 4: Konec.

Počítačové zpracování těchto dvou metod včetně výpočetních experimentů uvádí Kobzareva (2012).

4.2.1.2 Dynamická úloha

Doposud jsme předpokládali, že všechny požadavky jsou známé před vyjetím vozidla a není možné přijímat další objednávky na přepravu zásilek. Pro úlohu kurýrní služby je však typická opačná situace, neboť právě v tom spočívá smysl tohoto podnikání. Pro dynamické rozšíření úlohy DARP využil Psaraftis (1980) dynamické programování, Jaw et al. (1986) popsali heuristické metody. Berbeglia et al. (2012) uvádí využití constraint programming. Psaraftis (1983 a 1980) se zabývá dynamickou „many-to-many" verzí úlohy DARP. Uveďme re-optimalizační matematický model (Fábry, 2010) pro dynamickou úlohu kurýrní služby s jedním vozidlem:

$$\text{minimalizovat} \quad z = \sum_{i \in U_N} \sum_{j \in U_N} c_{ij} x_{ij}, \tag{4.19}$$

za podmínek

$$\sum_{j \in U_N} x_{ij} = 1, \quad i \in U_N, \tag{4.20}$$

$$\sum_{i \in U_N} x_{ij} = 1, \quad j \in U_N, \tag{4.21}$$

$$u_i - u_j + |U_N| x_{ij} \le |U_N| - 1, \quad i \in U_N, \quad j \in U_N - \{1\}, \quad i \ne j, \tag{4.22}$$

$$u_{i_{2k-1}} \le u_{i_{2k}}, \quad i_{2k-1}, i_{2k} \in \omega, \quad k = 1,2,...,\frac{|\omega|}{2} - 1, \tag{4.23}$$

$$u_1 = 0, \tag{4.24}$$

$$x_{1 j_{next}} = 1, \tag{4.25}$$

$$x_{ij} \in \{0,1\}, \quad i, j \in U_N, \tag{4.26}$$

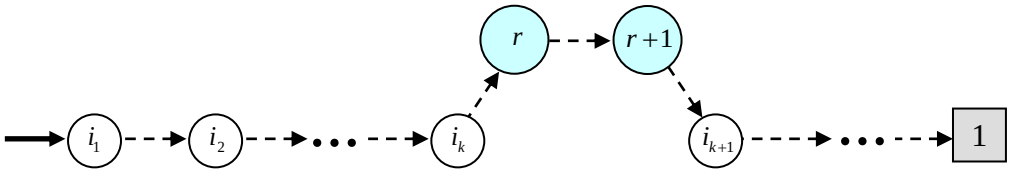
$$u_i \in R_0^+, \quad i \in U_N. \tag{4.27}$$

V modelu je nutné upravit původní matici nejkratších vzdáleností tak, že vzdálenost $c_{1 j_{next}}$ mezi výchozím místem č. 1 a místem j_{next}, ke kterému vozidlo směřuje v okamžiku vzniku nového požadavku, změníme na hodnotu odpovídající délce dosud absolvované trasy z výchozího místa do místa j_{next}. V množině U_N jsou místa, která musí vozidlo ještě navštívit, včetně místa j_{next}, výchozího místa č. 1, místa vyzvednutí nové zásilky a místa jejího doručení. Počet všech míst je dán hodnotou $|U_N|$. Tato množina obsahuje místa doručení dvojího typu: ty, jejichž zásilky nebyly dosud vyzvednuty, a ty, které čekají na doručení již vyzvednu-

tých zásilek. Nechť ω je rostoucí posloupnost indexů všech míst z množiny U_N, týkajících se dosud nevyzvednutých zásilek. Počet prvků této posloupnosti $|\omega|$ je vždy sudé číslo. Nerovnosti (4.23) jsou formulovány právě pro prvky této posloupnosti a zaručují, že v optimální trase bude zásilka nejprve vyzvednuta a teprve potom doručena.

Následující vkládací algoritmus (Fábry, 2010) předpokládá nalezení trasy pro statickou úlohu kurýrní služby, ať již za pomocí optimalizačního modelu (4.8) – (4.15) nebo některé heuristické metody. Při použití vkládacího algoritmu pro zařazení nového požadavku je nutné respektovat důležité omezení, které určuje, že místo doručení zásilky lze vložit na trase až za místo jejího vyzvednutí. Nechť $U_N = \{i_1, i_2, ..., i_m\}$ je posloupnost *m* míst ($i_m = 1$), která vozidlo musí ještě navštívit podle naplánované trasy. Vzhledem ke způsobu číslování míst, týkajících se jedné zásilky, můžeme označit indexem *r* uzel, v němž má být zásilka vyzvednuta a indexem $r+1$ uzel, v němž má být doručena. Mohou nastat dvě situace:

(1) Uzel *r* je vložen mezi uzly i_k a i_{k+1}, uzel $r+1$ pak bezprostředně za něj, tj. před uzel i_{k+1} (obr. 4.4).



Obr. 4.4 – Vložení míst vyzvednutí zásilky a jejího doručení bezprostředně za sebe

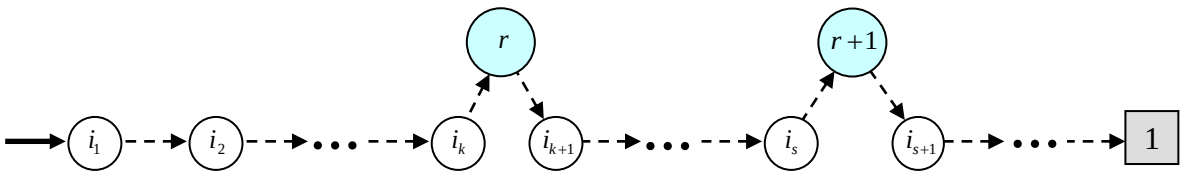
Prodloužení stávající trasy po vložení dvou nových míst bezprostředně za sebe lze pak vypočítat jako

$$\Delta z_k = c_{i_k, r} + c_{r, r+1} + c_{r+1, i_{k+1}} - c_{i_k, i_{k+1}}, \quad k = 1,2,...,m-1. \tag{4.28}$$

Cílem je nalézt takový index *t*, pro který platí:

$$\Delta z_t^1 = \min_{k=1,2,...,m-1} \Delta z_k . \tag{4.29}$$

(2) Uzel *r* je vložen mezi uzly i_k a i_{k+1}, uzel $r+1$ mezi místa i_s a i_{s+1}, přičemž prvním uzlem, za nějž může být uzel $r+1$ vložen, je zákazník i_{k+1} (obr. 4.5).



Obr. 4.5 – Vložení místa vyzvednutí zásilky a místa jejího doručení mezi různá místa na trase

Uzlem i_k nemůže být uzel i_{m-1}, neboť v tomto případě by se jednalo o předchozí situaci, kdy jsou uzly r a $r+1$ vloženy bezprostředně za sebe. Prodloužení trasy po vložení sídla nového zákazníka a místa doručení zásilky je

$$\Delta z_{ks} = c_{i_k,r} + c_{r,i_{k+1}} - c_{i_k,i_{k+1}} + c_{i_s,r+1} + c_{r+1,i_{s+1}} - c_{i_s,i_{s+1}}, \quad k = 1,2,...,m-2,\ s = k+1,k+2,...,m-1. \tag{4.30}$$

Cílem je nalézt takové indexy *t* a *p*, pro které platí:

$$\Delta z_{tp}^2 = \min_{\substack{k=1,2,...,m-2 \\ s=k+1,k+2,...,m-1}} \Delta z_{ks}\,. \tag{4.31}$$

Výběr nižší z hodnot (4.29) a (4.31) určí nejvýhodnější strategii zařazení nového požadavku na doručení zásilky do stávající trasy.

Zavedení časových oken významným způsobem přiblíží dynamickou úlohu kurýrní služby reálnému světu. Úprava re-optimalizačního modelu (4.19) – (4.27) je velice snadná, ovšem pro reálné úlohy, v nichž je zapotřebí okamžitě reagovat na nové požadavky, je tento přístup vzhledem k výpočetní složitosti NP-obtížných úloh, nevhodný. Stejně tak výše uvedený vkládací algoritmus není, v případě úlohy s jedním vozidlem při zavedení časových oken, vhodným postupem, neboť nebude snadné nalézt přípustné řešení. I kdyby takové řešení existovalo, bude zřejmě ve většině případů představovat velmi neefektivní trasu (vzhledem k minimalizaci vzdálenosti ujeté vozidlem). Bude tedy nutné použít jiný heuristický postup, vhodnější pro vytváření tras v úloze obchodního cestujícího s časovými okny. Lze použít i modifikaci zobecněného vkládacího algoritmu pro TSPTW (Gendreau et al., 1998a), která ovšem generuje zcela novou trasu z dosud nenavštívených uzlů, podobně jako re-optimalizační model.

4.2.2 Kapacitní úloha kurýrní služby s jedním vozidlem

V reálných úlohách se často stává, že velikost požadavků je vzhledem ke kapacitě vozidla významná. Nechť q_i je velikost zásilky, kterou je nutné vyzvednout v místě *i* a doručit do místa $i + 1$, kde $i = 2,4,..,2n$. Předpokládejme kapacitu vozidla o velikosti *V*. Na rozdíl od standardní rozvozní úlohy je pro vyřízení všech požadavků možné použít jediné vozidlo i v případě, že celková velikost všech požadavků převyšuje jeho kapacitu. Důvodem je skutečnost, že vozidlo na své trase velikost nákladu jednak zvyšuje (při vyzvednutí nějaké zásilky), ale také snižuje (při doručení zásilky). Matematický model statické úlohy lze pak definovat takto (Fábry, 2008):

$$\text{minimalizovat} \quad z = \sum_{i=1}^{2n+1} \sum_{j=1}^{2n+1} c_{ij} x_{ij}\,, \tag{4.32}$$

za podmínek

$$\sum_{j=1}^{2n+1} x_{ij} = 1, \quad i = 1,2,...,2n+1, \tag{4.33}$$

$$\sum_{i=1}^{2n+1} x_{ij} = 1, \quad j = 1,2,...,2n+1, \tag{4.34}$$

$$v_i + q_{2j} - V(1 - x_{i,2j}) \le v_{2j}, \quad i = 1,2,...,2n+1, \quad j = 1,2,...,n, \quad i \ne j\,, \tag{4.35}$$

$$v_i - q_{2j+1} - V(1 - x_{i,2j+1}) \le v_{2j+1}, \quad i = 1,2,...,2n+1, \quad j = 1,2,...,n, \quad i \ne j\,, \tag{4.36}$$

$$0 \le v_i \le V, \quad i = 2,3,...,2n+1, \tag{4.37}$$

$$u_i - u_j + (2n+1)x_{ij} \le 2n, \quad i = 1,2,...,2n+1, \quad j = 2,3,...,2n+1, \quad i \ne j\,, \tag{4.38}$$

$$u_{2i} \le u_{2i+1}, \quad i = 1,2,...,n\,, \tag{4.39}$$

$$u_1 = 0\,,\ v_1 = 0\,, \tag{4.40}$$

$$x_{ii} = 0, \quad i = 1,2,...,2n+1, \tag{4.41}$$

$$x_{ij} \in \{0,1\}, \quad i, j = 1,2,...,2n+1, \quad i \ne j\,, \tag{4.42}$$

$$u_i \in R_0^+, v_i \in R_0^+, \quad i = 2,3,...,2n+1\,. \tag{4.43}$$

Rovnice (4.33) a (4.34) zajišťují, že všechna místa (včetně výchozího) budou navštívena právě jednou. Podmínky (4.35), známé z modelu standardní rozvozní úlohy, se týkají bilance nákladů v uzlech, v nichž se zásilky vyzvedávají. Pokud se uskuteční přejezd do tohoto uzlu, zvýší se náklad vozidla[59] o velikost příslušného požadavku. Nerovnosti (4.36) jsou zavedeny z důvodu nastavení hodnot v_{2j+1}, které představují velikost nákladu po doručení zásilky v uzlu $2j + 1$. Podmínky (4.37) zabraňují překročení kapacity vozidla. Nerovnosti (4.38) a (4.39) jsou v modelu nutné pro eliminaci parciálních cyklů a pro zajištění základního předpokladu úlohy kurýrní služby, a sice nutnosti doručení zásilky až po jejím vyzvednutí.

4.2.3 Úloha kurýrní služby s několika vozidly

V této části se budeme zabývat úlohou, v níž je pro doručení zásilek použito více vozidel. Protože je použití několika vozidel vhodné i pro úlohu s časovými okny, uvedeme model úlohy, v níž jsou definovány nejdříve možné termíny vyzvednutí zásilek a nejpozději přípustné

[59] Jak bylo uvedeno v podkapitole 3.1, hodnota proměnné může být vyšší, než je skutečný náklad vozidla.

termíny jejich doručení (Fábry, 2008). Vzhledem k očíslování uzlů (v uzlech se sudým číslem jsou zásilky vyzvedávány a v uzlech s lichým číslem doručovány) je možné označit tento termín jako a_i pro všechny uzly (kromě výchozího místa), tedy pro $i = 2,3,...,2n+1$. V úloze je nutné znát doby přejezdů mezi uzly *i* a *j*, které jsou označeny t_{ij}. Ve výchozím místě se nachází *K* vozidel s různou kapacitou V_k $(k = 1,2,...,K)$. Cílem je tedy najít trasy těch vozidel, která z depotu vyjedou. K tomuto účelu zavedeme binární proměnnou x_{ij}^k, která nabývá hodnoty 1 v případě, že *k*-té vozidlo (vozidlo na *k*-té trase) bezprostředně po vyzvednutí či doručení zásilky v místě *i* vyzvedne či doručí zásilku v místě *j*. Matematický model lze pak zapsat následujícím způsobem:

$$\text{minimalizovat} \quad z = \sum_{k=1}^{K} \sum_{i=1}^{2n+1} \sum_{i=1}^{2n+1} c_{ij} x_{ij}^k \tag{4.44}$$

za podmínek

$$\sum_{k=1}^{K} \sum_{j=1}^{2n+1} x_{ij}^k = 1, \quad i = 2,3,...,2n+1, \tag{4.45}$$

$$\sum_{i=2}^{2n+1} x_{1i}^k \le 1, \quad k = 1,2,...,K, \tag{4.46}$$

$$\sum_{i=1}^{2n+1} x_{ij}^k = \sum_{l=1}^{2n+1} x_{jl}^k, \quad j = 1,2,...,2n+1, \; k = 1,2,...,K, \tag{4.47}$$

$$v_i^k + q_{2j} - V_k(1 - x_{i,2j}^k) \le v_{2j}^k, \quad i = 1,2,...,2n+1, \quad j = 2,3,...,n, \quad i \ne j, \quad k = 1,2,...,K, \tag{4.48}$$

$$v_i^k - q_{2j+1} - V_k(1 - x_{i,2j+1}^k) \le v_{2j+1}^k, \; i = 1,2,...,2n+1, \; j = 2,3,...,n, \; i \ne j, \quad k = 1,2,...,K, \tag{4.49}$$

$$0 \le v_i^k \le V_k, \quad i = 2,3,...,2n+1, \quad k = 1,2,...,K, \tag{4.50}$$

$$\tau_i^k + t_{ij} - M(1 - x_{ij}^k) \le \tau_j^k, \quad i = 1,2,...,2n+1, \quad j = 2,3,...,2n+1, \quad i \ne j, \quad k = 1,2,...,K, \tag{4.51}$$

$$\tau_{2i}^k \le \tau_{2i+1}^k, \quad i = 1,2,...,n, \quad k = 1,2,...,K, \tag{4.52}$$

$$\tau_{2i}^k \ge a_{2i} \sum_{j=1}^{2n+1} x_{2i,j}^k, \quad i = 1,2,...,n, \quad k = 1,2,...,K, \tag{4.53}$$

$$\tau_{2i+1}^k \le a_{2i+1} \sum_{j=1}^{2n+1} x_{2i+1,j}^k, \quad i = 1,2,...,n, \quad k = 1,2,...,K, \tag{4.54}$$

$$\sum_{j=1}^{2n+1} x_{2i,j}^{k} = \sum_{j=1}^{2n+1} x_{2i+1,j}^{k}, \quad i = 1,2,...,n, \; k = 1,2,...,K, \tag{4.55}$$

$$v_1^k = 0, \; \tau_1^k = 0, \quad k = 1,2,...,K, \tag{4.56}$$

$$x_{ii}^k = 0, \quad i = 1,2,...,2n+1, \quad k = 1,2,...,K, \tag{4.57}$$

$$x_{ij}^k \in \{0,1\}, \quad i, j = 1,2,...,2n+1, \quad i \neq j, \quad k = 1,2,...,K, \tag{4.58}$$

$$v_i^k \in R_0^+, \tau_i^k \in R_0^+, \quad i = 2,3,...,2n+1, \quad k = 1,2,...,K. \tag{4.59}$$

Rovnice (4.45) zajišťují, že každé místo (kromě depotu) je navštíveno právě jednou. Nerovnosti (4.46) povolují každému vozidlu v depotu maximálně jeden výjezd (tzn. buď vozidlo vůbec nevyjede, nebo vyjede pouze jednou). Podmínky (4.47) zaručí, že to samé vozidlo, které do některého uzlu vjede, z něj také vyjede. Nerovnosti (4.48) a (4.49) bilancují velikost nákladu vozidla po naložení či vyložení zásilky. Podmínky (4.50) zabrání překročení kapacity vozidel na všech trasách. Nerovnosti (4.51) jsou analogické nerovnostem v klasickém rozvozním problému s časovými okny, podmínky (4.53) a (4.54) definují dodržení nejdříve možných termínů vyzvednutí zásilek a nejpozději přípustných termínů pro jejich doručení. Doručení zásilky až po jejím vyzvednutí je opět zajištěna nerovnostmi (4.52). Rovnice (4.55) zajistí, že zásilku doručí totéž vozidlo, které ji vyzvedlo. Vzhledem ke složitosti matematického modelu této úlohy je pro ilustraci v Příloze 1 uveden jeho zápis v optimalizačním systému XPRESS-Ive[60], včetně datového souboru s jednoduchým ukázkovým problémem a jeho optimálním řešením.

Také u tohoto typu úloh lze uvažovat dynamické rozšíření. Experimenty s výše uvedeným modelem ukázaly, že již při větším rozsahu úlohy je výpočet v praxi nerealizovatelný. Jak bylo již několikrát v této práci uvedeno, v případě dynamických úloh je nutné reagovat okamžitě na nové požadavky v podobě změny tras vozidel. Proto opět hrají důležitou roli heuristické algoritmy (Fábry a Kobzareva, 2012). Uveďme algoritmy pro statickou nekapacitní úlohu bez časových oken, které lze rozšířit na úlohu, jejíž matematický model je uveden výše.

Modifikovaná metoda nejbližšího souseda

Opět budeme předpokládat jedno výchozí místo (označené indexem 1), v němž se nachází K vozidel. Množina U_k bude obsahovat dosud nezařazené uzly, které může v daném kroku navštívit vozidlo k, tj. na začátku budou všechny tyto množiny obsahovat všechny sudé uzly,

[60] Jedná se o produkt firmy FICO [129].

představující místa vyzvednutí zásilek. Nechť U_k^* je posloupnost míst, odpovídající vygenerované trase *k*-tého vozidla, j_k je poslední uzel na této trase a z_k je délka této trasy. Hodnota *z* představuje celkovou délku všech tras.

Krok 1: Pro $k = 1,2,...,K$ nastav $U_k = \{2,4,...,2n\}; U_k^* = \{1\};\ j_k = 1; z_k = 0$.
$z = 0.$

Krok 2: Najdi vozidlo *m* a místo *s* takové, že $c_{j_m s} = \min_{k=1,2,...,K} \min_{r \in U_k} c_{j_k r}$.
$U_m^* = U_m^* + \{s\};\ z_m = z_m + c_{j_m s};\ U_k = U_k - \{s\}$ pro $k = 1,2,...,K$.
Jestliže *s* je sudé, pak $U_m = U_m + \{s+1\}$.
$j_m = s$.

Krok 3: Když je $U_k = \emptyset$ pro $k = 1,2,...,K$, pak jdi na krok 4, jinak jdi na krok 2.

Krok 4: Pro $k = 1,2,...,K$ proveď:
Pokud $U_k^* \neq \{1\}$, pak $U_k^* = U_k^* + \{1\};\ z_k = z_k + c_{j_k 1}$.
$z = \sum_{k=1}^{K} z_k$.
Konec.

Modifikovaná vkládací metoda

Následující algoritmus je definovaný pro *K* vozidel, která budou všechna použita, tj. předpokládáme *K* okruhů. V množině U_k se nacházejí dosud nezařazené uzly, které lze v daném kroku zařadit do trasy *k*, tj. na začátku budou všechny tyto množiny obsahovat uzly $2,3,...,2n+1$. Nechť U_k^* je posloupnost míst, odpovídající vygenerované trase *k*-tého vozidla, u_i^k je prvek posloupnosti U_k^* na *i*-té pozici, z_k je délka *k*-té trasy. Hodnota *z* představuje celkovou délku všech tras.

Krok 1: Pro $k = 1,2,...,K$ nastav $U_k = \{2,3,...,2n+1\}$.
Pro $k = 1,2,...,K$ opakuj:
$c_{1s} = \max_{r \in U_k} c_{1r};\ U_k^* = \{1,s,1\};\ z_k = c_{1s} + c_{s1};\ U_k = U_k - \{s\}$
pro $i = 1,2,...,K, i \neq k$ proveď:
$U_i = U_i - \{s\}$
jestliže *s* je sudé, pak $U_i = U_i - \{s+1\}$, jinak $U_i = U_i - \{s-1\}$.

Krok 2: $\Delta z = \infty$.

Pro $k = 1,2,...,K$ opakuj:

pro každé $r \in U_k$ opakuj:

a) pokud je *r* sudé a v posloupnosti U_k^* existuje uzel *r* + 1 na *s*-té pozici, pak

$$\Delta z_{rj} = \min_{i=1,2,...,s-1} (c_{u_i^k, r} + c_{r, u_{i+1}^k} - c_{u_i^k, u_{i+1}^k}),$$

b) pokud je *r* liché a v posloupnosti U_k^* existuje uzel *r* – 1 na *s*-té pozici, pak

$$\Delta z_{rj} = \min_{i=s,s+1,...,|U_k^*|-1} (c_{u_i^k, r} + c_{r, u_{i+1}^k} - c_{u_i^k, u_{i+1}^k}),$$

c) jinak

$$\Delta z_{rj} = \min_{i=1,2,...,|U_k^*|-1} (c_{u_i^k, r} + c_{r, u_{i+1}^k} - c_{u_i^k, u_{i+1}^k}).$$

Pokud $\Delta z_{rj} < \Delta z$, pak

$\Delta z = \Delta z_{rj}$; *v* = *r*; *w* = *j*; *m* = *k*.

Do posloupnosti U_m^* přidej uzel *v* za prvek u_w^m.

$U_m = U_m - \{v\}$; $z_m = z_m + \Delta z$; $z = z + \Delta z$.

Pro $k = 1,2,...,K, k \neq m$ opakuj:

$U_k = U_k - \{v\}$

pokud je *v* sudé, pak $U_k = U_k - \{v+1\}$, jinak $U_k = U_k - \{v-1\}$.

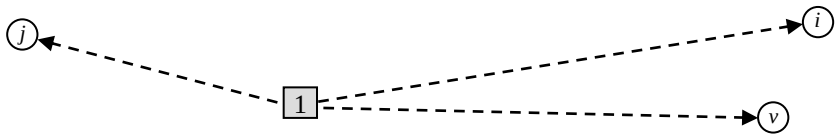
Krok 3: Když je $U_k = \emptyset$ pro $\forall k = 1,2,...,K$, pak jdi na krok 4, jinak jdi na krok 2.

Krok 4: Konec.

Tato modifikace předpokládá, že do obsluhy zákazníků se zapojí všechna vozidla, neboť v Kroku 1 se vytvoří *K* jednoduchých okruhů (přímých jízd typu 1-*s*-1), obsahujících výchozí místo a nějaký další uzel. V tom případě tedy předpokládáme, že počet $K \leq n$. Při takové podmínce je ovšem zcela evidentní, že tento algoritmus může být neefektivní z hlediska získání dobrého přípustného řešení (pokud se *K* bude blížit k *n*). Metoda tedy vyhovuje spíše úlohám, v nichž počet zásilek výrazně převyšuje hodnotu *K*, případně kdy připouštíme možnost, že pro přepravu nebudou použita všechna disponibilní vozidla. V Kroku 1 jsou navíc uzly vybírány na základě jednoduchého principu. Pro každé vozidlo je vždy vybrán uzel nejvzdálenější od výchozího místa. Na obr. 4.6 je uveden příklad demonstrující úskalí tohoto principu. Od výchozího místa č. 1 je nejvzdálenějším *i*-tý uzel[61], proto trasa prvního vozidla dle výše uvedeného Kroku 1 by byla $\{1,i,1\}$, trasa druhého vozidla pak $\{1,v,1\}$, ačkoli je evidentní, že z hlediska celkové efektivnosti je výhodnější poslat druhé vozidlo „opačným smě-

[61] Pro vysvětlení nejsou další uzly důležité.

rem“, tedy do uzlu *j*. Uzel *v* by totiž mohl být později případně zařazen do trasy prvního vozidla.



Obr. 4.6 – Vkládací algoritmus, inicializace tras

Jisté zlepšení tohoto inicializačního kroku lze najít v práci Svobodové (2013), která popisuje postup v případě dvou a tří vozidel, který lze samozřejmě zobecnit na *K* vozidel. Pokud jsou k dispozici tři vozidla, bude nejprve vybrán uzel *i* nejvzdálenější k výchozímu uzlu. Pak pro další dvě vozidla budou vybrány takové dva uzly *j* a *v*, pro které je součet $c_{ij}+c_{jv}+c_{vi}$ maximální. Alternativní možnost inicializace pro *K* vozidel je následující:

Krok 1: Pro $k=1,2,...,K$ nastav $U_k=\{2,3,...,2n+1\}$.

$c_{1s}=\max\limits_{v\in U_1} c_{1v}$; $U_1^*=\{1,s,1\}$; $z_1=c_{1s}+c_{s1}$; $U_1=U_1-\{s\}$.

Pro $k=2,3...,K$ proveď:

$U_k=U_k-\{s\}$

jestliže *s* je sudé, pak $U_k=U_k-\{s+1\}$, jinak $U_k=U_k-\{s-1\}$.

Pro $k=2,3...,K$ opakuj:

$\sum\limits_{p\le k-1} c_{i_2^p s}=\max\limits_{v\in U_k}\sum\limits_{p\le k-1} c_{i_2^p v}$; $U_k^*=\{1,s,1\}$; $z_k=c_{1s}+c_{s1}$; $U_k=U_k-\{s\}$

pro $i=1,2,...,K, i\neq k$ proveď:

$U_i=U_i-\{s\}$

jestliže *s* je sudé, pak $U_i=U_i-\{s+1\}$, jinak $U_i=U_i-\{s-1\}$.

Nejprve je nalezen uzel, který je nejvzdálenější výchozímu uzlu a ten je zařazen do trasy prvního vozidla, tj. vytvořena posloupnost $U_1^*=\{1,s,1\}$. Následně se vytvářejí trasy dalších vozidel tak, že se hledá takový uzel, který může být přiřazen danému vozidlu a zohledňují se nejkratší vzdálenosti (resp. jejich součet) tohoto uzlu od uzlů již přiřazených předchozím vozidlům. Index i_2^p označuje v pořadí vždy druhý uzel na trase *p*-tého vozidla, kde *p* musí být menší než *k* (trasy vozidel jsou vytvářeny postupně podle jejich pořadí). Pro nalezení *K* uzlů, které budou zařazeny do jednoduchých (přímých) tras, lze použít i následující optimalizační bivalentní model. V modelu předpokládáme symetrickou matici nejkratších vzdáleností:

$$\text{minimalizovat} \quad z = \sum_{i=1}^{2n} \sum_{j=i+1}^{2n+1} c_{ij} y_{ij} \tag{4.60}$$

za podmínek

$$y_{ij} \geq x_i + x_j - 1, \quad i = 1,2,...,2n,\ j = i+1, i+2,....,2n+1, \tag{4.61}$$

$$x_{2i} + x_{2i+1} \leq 1, \quad i = 1,2,...,n, \tag{4.62}$$

$$\sum_{i=2}^{2n+1} x_i = K, \tag{4.63}$$

$$x_1 = 1, \tag{4.64}$$

$$x_i \in \{0,1\}, \quad i = 2,3,...,2n+1, \tag{4.65}$$

$$y_{ij} \in \{0,1\}, \quad i = 1,2,...,2n,\ j = i+1, i+2,...,2n+1. \tag{4.66}$$

V modelu jsou zavedeny bivalentní proměnné dvojího typu. Proměnná x_i je rovna 1, jestliže uzel *i* je vybrán do některé z *K* tras, jinak nabývá hodnoty 0. Proměnná y_{ij} je rovna 1, pokud je brána v úvahu nejkratší vzdálenost mezi zařazenými uzly *i* a *j*. Účelová funkce (4.60) představuje součet všech vzdáleností mezi vybranými uzly. Nerovnice (4.61) zajišťují nastavení proměnné y_{ij} na hodnotu 1 v případě, že obě proměnné x_i a x_j budou rovny jedné, neboli pokud budou příslušné uzly vybrány, započítá se v účelové funkci jejich nejkratší vzdálenost. Omezující podmínky (4.62) zabrání tomu, aby na dvou trasách byly vybrány uzly týkající se jedné zásilky. Pokud je vybrán uzel, v němž se nějaká zásilka vyzvedává, nemůže být místo doručení vybráno do žádné jiné trasy, a naopak. Protože se má vytvořit *K* tras, musí být vybráno *K* uzlů, což zajišťuje rovnice (4.63). Nastavením podmínky (4.64) určíme, že výchozí místo bude vybráno, což je alternativou k původní inicializaci, v níž jsme hledali nejvzdálenější uzel od depotu. V tomto případě zohledňujeme vzdálenosti od výchozího uzlu v účelové funkci.

Pro vkládací algoritmus je inicializace tras poměrně důležitá a hraje významnou roli pro další část algoritmu. Jistě lze najít další možnosti, jak tento krok zefektivnit, což může být předmětem dalšího výzkumu. Další efektivní způsob je popsán v následující části, týkající se úlohy kurýrní služby s více kurýry umístěnými v různých depotech.

Oba modifikované heuristické algoritmy, tedy metoda nejbližšího souseda i vkládací metoda, se řadí mezi generující postupy, jejichž aplikací získá řešitel několik tras. Samozřejmě se nabízí otázka, zda je možné získané řešení zlepšit. V podkapitole 2.7 byl popsán algoritmus

zlepšující trasu nazvaný metoda výměn. Původní algoritmus byl koncipován pro výměnu hran v rámci jedné trasy, my jej modifikujeme pro úlohu kurýrní služby s několika vozidly. Ve skutečnosti nepůjde o výměnu hran, ale uzlů[62].

Modifikovaná metoda výměn

Předpokládejme K tras naplánovaných metodou nejbližšího souseda či vkládacím algoritmem. Nechť U_k^* je posloupnost míst, odpovídající vygenerované trase k-tého vozidla, u_i^k je prvek posloupnosti U_k^* na i-té pozici, z_k je délka k-té trasy, z je celková délka všech tras.

Krok 1: Pro každou trasu najdeme zásilku, tj. dvojici příslušných uzlů vyzvednutí a doručení, která přinese maximální redukci délky trasy, pokud tuto zásilku z trasy vyřadíme. Tato zásilka je kandidátem na výměnu. Označme tuto redukci jako $\Delta zExcl_k$ $(k = 1,2,...,K)$. Z těchto hodnot pak vybereme maximální redukci:

$$\Delta zExcl_m = \max_{k=1,2,...,K} \Delta zExcl_k .$$

Nechť tato redukce v trase m odpovídá dvěma uzlům r a $r + 1$ na pozicích v a w v posloupnosti U_m^*.

Krok 2: Pro všechny trasy $k = 1,2,...,K, k \neq m$ najdeme nejvýhodnější vložení uzlů r a $r + 1$ s využitím vkládacího algoritmu pro dynamickou úlohu, který byl popsán v části 4.2.1.2. U každé trasy tedy určíme její minimální prodloužení, které označíme $\Delta zExt_k$ $(k = 1,2,...,K, k \neq m)$. Z těchto hodnot dále vybereme minimální hodnotu:

$$\Delta zExt_p = \min_{\substack{k=1,2,...,K \\ k \neq m}} \Delta zExt_k .$$

Nechť toto prodloužení trasy p odpovídá vložení uzlu r za uzel u_i^p a uzlu $r + 1$ za uzel u_j^p. Je zřejmé, že $i \leq j$[63].

Krok 3: Jestliže $\Delta zExcl_m > \Delta Ext_p$, proveď přesun, tedy úpravu tras m a p, a pokračuj krokem 1, jinak jdi na krok 4.

Krok 4: Konec.

[62] Samozřejmě na tuto úlohu může být aplikován původní princip, tedy výměna hran v rámci jednoho okruhu, případně výměna hran napříč okruhy. Je ovšem nutné mít stále na paměti zachování chronologie vyzvednutí a doručení, týkající se téže zásilky.

[63] V případě $i = j$ budou oba uzly vloženy bezprostředně za sebe.

U tohoto algoritmu je zapotřebí upřesnit, že se ve skutečnosti nejedná a výměnu, ale přeřazení zásilky z jedné trasy do druhé. Výpočetní experimenty (Fábry a Kobzareva, 2012) ukazují, že dodatečná změna vygenerovaných heuristických řešení přináší výrazné úspory. I v případě tohoto zlepšujícího postupu lze najít další modifikace. Pokud by v Kroku 3 algoritmus skončil, neboť by plánované přeřazení nepřineslo žádoucí efekt, je možné se vrátit ke Kroku 1 a pokusit se o efektivní přeřazení z druhé nejvýhodnější trasy v pořadí, tedy trasu *m* zařadíme do jakéhosi „tabu seznamu“. Takto můžeme postupovat i s dalšími trasami.

Tento postup je výhodné využít v úlohách, v nichž uvažujeme kapacitu vozidel či časová okna. Přeřazování zásilek je užitečné i v případě nerovnoměrného zatížení jednotlivých vozidel z hlediska počtu vyřízených zásilek, případně z hlediska časové délky jednotlivých tras a délky pracovního dne.

4.2.4 Úloha kurýrní služby s několika vozidly v několika výchozích místech

Následující text je věnován úlohám, v nichž jsou vozidla kurýrní služby umístěna v oddělených depotech. Uvedeme úpravu heuristických algoritmů vytvořených pro úlohu s vozidly ve společném výchozím místě, které byly prezentovány výše. Bez újmy na obecnosti předpokládejme existenci lichého počtu výchozích míst *K*, a to z důvodu předem zavedeného značení sudých míst vyzvednutí zásilek a lichých míst jejich doručení. V případě, že v úloze existuje sudý počet výchozích míst, zavedeme fiktivní místo. V každém depotu je umístěno jedno vozidlo. Depoty se nacházejí v uzlech $k = 1,2,...,K$. Uzly $K+1, K+3,..., K+2n-1$ pak představují místa vyzvednutí zásilek a uzly $K+2, K+4,..., K+2n$ místa jejich doručení. Opět platí předpoklad, že zásilka, která je vyzvednuta v uzlu i, je doručena do uzlu $i+1$.

Modifikovaná metoda nejbližšího souseda

V následujícím algoritmu budeme opět předpokládat, že vyjedou všechna vozidla, tj. postupně bude vygenerováno K tras[64]. Podobně jako v případě více vozidel vyjíždějících z jednoho výchozího místa, množina U_k bude na začátku obsahovat všechny sudé uzly, představující místa vyzvednutí zásilek. Nechť U_k^* je posloupnost míst, odpovídající vygenerované

[64] V případě zavedení fiktivního depotu bude ve všech krocích příslušný kurýr ignorován, tj. nebude pro něj žádná trasa inicializována.

trase *k*-tého vozidla, j_k je poslední uzel na této trase a z_k je délka této trasy. Hodnota *z* představuje celkovou délku všech tras.

Krok 1: Pro $k = 1,2,...,K$ nastav $U_k = \{K+1, K+3, ..., K+2n-1\}$.

Krok 2: Pro $k = 1,2,...,K$ opakuj

$c_{ks} = \min_{r \in U_k} c_{kr}$

$U_k^* = \{k, s\}$; $z_k = c_{ks}$

pro $l = 1,2,...,K$ nastav $U_l = U_l - \{s\}$

$U_k = U_k + \{s+1\}$

$j_k = s$.

Krok 3: $c_{j_m s} = \min_{k=1,2,...,K} \min_{r \in U_k} c_{j_k r}$

$U_m^* = U_m^* + \{s\}$; $z_m = z_m + c_{j_m s}$; $j_m = s$

Pro $k = 1,2,...,K$ nastav $U_k = U_k - \{s\}$.

Pokud *s* je sudé, pak nastav $U_m = U_m + \{s+1\}$.

Krok 4: Když je $U_k = \emptyset$ pro $\forall k = 1,2,...,K$, pak jdi na krok 5, jinak jdi na krok 3.

Krok 5: Pro $k = 1,2,...,K$ nastav $U_k^* = U_k^* + \{k\}$; $z_k = z_k + c_{j_k k}$

$z = \sum_{k=1}^{K} z_k$.

Konec.

V Kroku 2 je postupně pro každý depot nalezen nejbližší uzel, v němž je možné vyzvednout zásilku. Tento krok může být modifikován např. tak, že se nejprve nalezne uzel, který je nejblíž k jakémukoli depotu. Tento uzel je pak přiřazen odpovídajícímu vozidlu. Takto jsou potupně inicializovány všechny trasy. Ve skutečnosti je toto pravidlo dále použito i v Kroku 3 při rozšiřování tras hledáním nejbližšího souseda k poslednímu uzlu jakékoli trasy. Zařazení určitého uzlu do trasy vozidla *m* znamená jeho vyřazení ze všech množin U_k a v případě, že se jedná o uzel, v němž se vyzvedává zásilka, je navíc pro dané vozidlo do množiny U_m zařazeno příslušné místo doručení této zásilky.

Výpočetní experimenty s navrženým algoritmem ukazují, že některé trasy mohou být mnohem delší než trasy ostatních vozidel. V takovém případě je možné zavést určitá pravidla, na jejichž základě získáme trasy srovnatelné délky, ovšem za cenu mnohdy výrazného zhoršení výsledku z hlediska celkové délky všech tras. Značná nerovnoměrnost rozvržení délek tras může být do jisté míry způsobena i tím, že při generování tras nebereme ohled na to, zda

je dané vozidlo schopné příslušnou trasu absolvovat např. během běžné pracovní doby. V následující úpravě algoritmu je zaveden parametr *Tmax* představující časový limit, který musí splnit každé vozidlo. V úloze je nutné znát doby přejezdů mezi uzly *i* a *j*, které jsou označeny t_{ij}. Dále pro každé vozidlo $k = 1,2,...,K$ zavedeme proměnnou T_k, jejíž hodnota udává celkovou dobu jízdy vozidla na trase *k* v daném kroku vytváření trasy. Pokud označíme počet uzlů na trase $h_k = \left|U_k^*\right|$, pak lze psát

$$T_k = \sum_{i=1}^{h_k - 1} d_{u_i, u_i + 1}, \tag{4.67}$$

kde u_i je číslo uzlu na *i*-té pozici posloupnosti U_k^*. Zřejmě je $u_1 = k$ a $u_{h_k} = j_k$.

Protože během generování trasy metodou nejbližšího souseda mohou existovat některá dosud nenavštívená místa doručení, spojená s místy vyzvednutí, která jsou již v trase zahrnuta, je třeba s návštěvou těchto míst počítat a zahrnout odpovídající dobu do celkové délky trasy tak, abychom do trasy nezařadili uzly, jejichž návštěva by ohrozila splnění časového limitu *Tmax*. K tomuto účelu je vhodné zavést množinu všech povinných, dosud nenavštívených, míst doručení, kterou označíme B_k. Upravený algoritmus pak lze zapsat následovně:

Krok 1: Pro $k = 1,2,...,K$ nastav $U_k = \{K+1, K+3, ..., K+2n-1\}$.

Krok 2: Pro $k = 1,2,...,K$ opakuj

$c_{ks} = \min_{r \in U_k} c_{kr}$

$U_k^* = \{k, s\}$; $z_k = c_{ks}$; $T_k = t_{ks}$

pro $l = 1,2,...,K$ nastav $U_l = U_l - \{s\}$

$U_k = U_k + \{s+1\}$; $B_k = \{s+1\}$

$j_k = s$.

Krok 3: Pro $k = 1,2,...,K$ opakuj

pro $\forall i \in U_k$ opakuj

$U_k' = U_k$; pokud *r* je sudé pak $U_k' = U_k' + \{r+1\}$

$B_k' = B_k$; pokud *r* je sudé pak $B_k' = B_k' + \{r+1\}$

$T_k' = T_k$; $r = i$

opakuj dokud je $B_k' \neq \emptyset$

$t_{rv} = \min_{j \in B_k'} c_{rj}$; $T_k' = T_k' + d_{rv}$; $B_k' = B_k' - \{v\}$; $r = v$

$T_k' = T_k' + t_{rk}$

pokud $T_k' >$ *Tmax* pak $U_k' = U_k' - \{i\}$

$c_{j_m s} = \min_{k=1,2,...,K} \min_{r \in U_k'} c_{j_k r}$

$U_m^* = U_m^* + \{s\};\ z_m = z_m + c_{j_m s};\ T_m = T_m + t_{j_m s};\ j_m = s$

pokud *s* je sudé pak

pro $k = 1,2,...,K$ nastav $U_k = U_k - \{s\}$

$U_m = U_m + \{s+1\};\ B_m = B_m + \{s+1\}$

jinak

$U_m = U_m - \{s\}$.

Krok 4: Když je $U_k = \emptyset$ pro $\forall\, k = 1,2,...,K$, pak jdi na krok 5, jinak jdi na krok 3.

Krok 5: Pro $k = 1,2,...,K$ nastav $U_k^* = U_k^* + \{k\};\ z_k = z_k + c_{j_k k}$

$z = \sum_{k=1}^{K} z_k$.

Konec.

Hlavní změna algoritmu je v Kroku 3, v němž uzel $i \in U_k$ může být vybrán jako nejbližší soused pro rozšíření trasy *k*-tého vozidla jen v případě, že bude možné dokončit trasu během časového limitu *Tmax*. Je nutné zohlednit i situaci, že bude do trasy vybrán sudý uzel *i*, a tudíž je zapotřebí počítat i s nutnou návštěvou uzlu *i* + 1. Algoritmus ovšem neuvažuje případy, v nichž žádné vozidlo není schopné vyřídit zásilku vzhledem k zadanému časovému limitu. Samozřejmě je možné algoritmus dále modifikovat tak, aby minimalizoval výskyt těchto negativních situací. Nicméně dokonce ani použití optimalizačního modelu nemusí zaručit, že všechny zásilky bude možné vyřídit během dne se stávajícím vozovým parkem.

Modifikovaná vkládací metoda

Jak bylo uvedeno v předchozí části, je inicializace tras u vkládací metody, aplikované na úlohu s několika vozidly, komplikovanější než v problémech s jedním vozidlem. Nyní je navíc nutné zohlednit fakt, že vozidla nevyjíždějí ze stejného depotu, ale že jejich stanoviště jsou různá. Inicializace tras je tudíž mnohem složitější a samozřejmě existuje celá řada možných přístupů k tomuto, pro vkládací metodu velice důležitému, kroku. V navržené metodě navíc, stejně jako v předchozí modifikaci algoritmu nejbližšího souseda, uvažujeme časový limit *Tmax* pro jednotlivé trasy.

Inicializace každé trasy spočívá v nalezení vhodného uzlu, přes který bude vytvořen jednoduchý cyklus. V něm bude kromě vybraného uzlu zahrnutý také druhý uzel související s danou zásilkou. Pro každé vozidlo *k* hledáme uzel, který je blíže k jeho depotu než k depotu ostatních vozidel (viz proměnné *LowDist* a *LowLoc* v níže uvedeném algoritmu). Jestliže žád-

ný takový uzel neexistuje, tj. všechny uzly jsou dále k depotu *k* než k některému dalšímu depotu, je jako nejvhodnější vybrán uzel s nejmenším rozdílem mezi jeho vzdáleností od depotu *k* a minimální vzdáleností od ostatních depotů (viz proměnné *UpDif* a *UpLoc*). V algoritmu je použito předchozí značení; navíc označme $u_i^k \in U_k^*$ číslo uzlu který je na *i*-té pozici trasy *k*-tého vozidla.

Krok 1: Pro $k = 1,2,...,K$ nastav $U_k = \{K+1, K+2, ..., K+2n\}$.

Krok 2: Pro $k = 1,2,...,K$ opakuj

$LowDist = -\infty$; $UpDif = +\infty$

pro $\forall\, i \in U_k$ opakuj

pokud $c_{ki} \le \min\limits_{\substack{l=1,2,...,K \\ l \ne k}} c_{li}$ pak

pokud $c_{ki} > LowDist$ pak

$LowDist = c_{ki}$; $LowLoc = i$

jinak pokud $(c_{ki} - \min\limits_{\substack{l=1,2,...,K \\ l \ne k}} c_{li}) < UpDif$ pak

$UpDif = (c_{ki} - \min\limits_{\substack{l=1,2,...,K \\ l \ne k}} c_{li})$; $UpLoc = i$

pokud $LowDist > -\infty$ pak $s = LowLoc$ jinak $s = UpLoc$

pokud *s* je sudé pak

$U_k^* = \{k, s, s+1, k\}$; $z_k = c_{ks} + c_{s,s+1} + c_{s+1,k}$; $T_k = t_{ks} + t_{s,s+1} + t_{s+1,k}$

jinak

$U_k^* = \{k, s-1, s, k\}$; $z_k = c_{k,s-1} + c_{s-1,s} + c_{sk}$; $T_k = t_{k,s-1} + t_{s-1,s} + t_{sk}$

$h_k = 4$

pro $l = 1,2,...,K$ opakuj

$U_l = U_l - \{s\}$.

pokud *s* je sudé pak $U_l = U_l - \{s+1\}$ jinak $U_l = U_l - \{s-1\}$.

Krok 3: $\Delta z = +\infty$

pro $k = 1,2,...,K$ opakuj

pro všechna sudá $r \in U_k$ opakuj

$$\Delta zImm_{rj} = \min_{i=1,2,...,h_k-1} (c_{u_i^k r} + c_{r,r+1} + c_{r+1,u_{i+1}^k} - c_{u_i^k u_{i+1}^k})$$

$$\Delta tImm = t_{u_j^k r} + t_{r,r+1} + t_{r+1,u_{j+1}^k} - t_{u_j^k u_{j+1}^k}$$

$$\Delta zLat_{rvw} = \min_{\substack{i=1,2,...,h_k-2 \\ m=i+1,i+2,...,h_k-1}} (c_{u_i^k r} + c_{r u_{i+1}^k} - c_{u_i^k u_{i+1}^k} + c_{u_m^k,r+1} + c_{r+1,u_{m+1}^k} - c_{u_m^k u_{m+1}^k})$$

$$\Delta tLat = t_{u_i^k r} + t_{r u_{i+1}^k} - t_{u_i^k u_{i+1}^k} + t_{u_m^k,r+1} + t_{r+1,u_{m+1}^k} - t_{u_m^k u_{m+1}^k}$$

pokud $T_k + \Delta tImm < Tmax$ a $\Delta zImm_{rj} < \Delta z$ pak

Imm =*PRAVDA*; *Msg* = *k*; *Pick* = *r*; *After1* = *j*;
$\Delta z = \Delta zImm_{rj}$; $\Delta T = \Delta tImm$

pokud $T_k + \Delta tLat < Tmax$ a $\Delta zLat_{rvw} < \Delta z$ pak
Imm =*NEPRAVDA*; *Msg* = *k*; *Pick* = *r*; *After1* = *v*; *After2* = *w*
$\Delta z = \Delta zLat_{rvw}$; $\Delta T = \Delta tLat$

pokud $Imm = PRAVDA$ a $\Delta z < +\infty$ pak

vlož uzly *Pick* a *Pick* + 1 do posloupnosti U^*_{Msg} na pozice *After1* + 1 a *After1* + 2

jinak pokud $\Delta z < +\infty$ pak

vlož uzly *Pick* a *Pick* + 1 do posloupnosti U^*_{Msg} na pozice *After1* + 1 a *After2* + 1

pokud $\Delta z < +\infty$ pak

$h_{Msg} = h_{Msg} + 2$; $z_{Msg} = z_{Msg} + \Delta z$; $T_{Msg} = T_{Msg} + \Delta T$

pro $k = 1,2,...,K$ opakuj
$U_k = U_k - \{Pick\}$; $U_k = U_k - \{Pick + 1\}$.

Krok 4: Když je $U_k = \emptyset$ pro $\forall\, k = 1,2,...,K$, pak jdi na krok 5, jinak jdi na krok 3.

Krok 5: $z = \sum_{k=1}^{K} z_k$.

Konec.

V každé iteraci jsou vloženy do některé trasy dva uzly, ve výše uvedeném algoritmu označené *Pick* a *Pick* + 1. Existují dvě možnosti jejich vložení:

1) oba dva uzly jsou vloženy bezprostředně za sebe,

2) mezi oběma uzly bude několik uzlů, již dříve vložených do trasy.

V prvním případě hodnota $\Delta zImm_{rj}$ vyjadřuje minimální prodloužení současné trasy vozidla *k*, jestliže uzel vyzvednutí *r* a uzel doručení *r* + 1 jsou společně vloženy za uzel na pozici *j* posloupnosti U^*_k. Hodnota $\Delta tImm$ je počítána pro ověření, zda je přípustné tímto způsobem vložit dané uzly z hlediska časového limitu stanoveného pro trasy. Podobně ve druhém případě hodnota $\Delta zLat_{rvw}$ odpovídá minimálnímu prodloužení trasy, jestliže uzly *r* a *r* + 1 jsou do posloupnosti U^*_k vloženy odděleně, a sice uzel *r* na pozici *v* a uzel *r* + 1 na pozici *w*. Opět je ověřena přípustnost této možnosti z hlediska časového omezení, tentokrát pomocí hodnoty $\Delta tLat$.

Po nalezení všech přípustných možností pro všechny trasy je určeno nejvýhodnější vložení uzlů *Pick* a *Pick* + 1 do trasy vozidla *Msg* do posloupnosti U^*_{Msg} za pozici *After1* (pokud *Imm* = *PRAVDA*) nebo za pozice *After1* a *After2* (pokud *Imm* = *NEPRAVDA*). Všechny množiny U_k jsou nakonec příslušným způsobem změněny, tj. ze všech množin jsou odstraněny uzly *Pick* a *Pick* + 1. Podobně jako předešlý algoritmus nejbližšího souseda, také tato metoda neuvažuje situaci, kdy zásilka nemůže být vložena do žádné trasy kvůli nesplnění časového limitu.

Na generované trasy vozidel lze aplikovat modifikovanou metodu výměn, která je v podstatě totožná s metodou popsanou v předchozí části pro trasy začínající a končící v jednom depotu. Pokud je zaveden časový limit *Tmax*, pak je samozřejmě nutné při přesunu zásilky z jedné trasy do jiné tuto hodnotu respektovat.

Výpočetní experimenty

Přibylová (2014) ve své práci vytvořila aplikace ve VBA for Excel na základě výše uvedených metod a provedla výpočetní experimenty se zajímavými závěry, jejichž statisticky významné zobecnění by samozřejmě vyžadovalo důkladnější analýzu a další experimenty provedené na různých datových souborech. Zmíněné experimenty se týkaly reálné matice nejkratších vzdáleností mezi 650 místy, z nichž byly generovány případy s 10, 20, 30 a 40 zásilkami pro 3, 5, 7 a 11 vozidel. Z experimentů lze mj. odhadnout, že přibližně v 75 % případů poskytuje vkládací metoda lepší výsledky než metoda nejbližšího souseda. Při použití metody výměn bylo 70 % řešení získaných metodou nejbližšího souseda vylepšeno, zatímco u metody vkládací se jednalo jen o 15 %. I po této aplikaci zlepšující heuristiky bylo stále 65 % lepších řešení získáno vkládací metodou. Aplikace metody výměn přinesla zkrácení tras maximálně o 5 % celkové délky. Další úsporu by zřejmě přinesly další modifikace této metody.

Všechny uvedené algoritmy lze bez většího úsilí modifikovat pro dynamické rozšíření úloh s několika vozidly, což je ovšem již nad rámec této práce.

4.3 Zobecněný PDP

Přestože český název pro následující typ úlohy neexistuje, je možné v této práci používat toto označení, neboť do jisté míry vystihuje podstatu problému. Jak bylo uvedeno v úvodu této kapitoly, existují problémy typu „one-to-many-to-one“, v nichž každé místo může být místem

nakládky i vykládky zároveň. To je odlišnost např. od Transhipment problému, popsaného výše, v němž každé místo je buď místem vykládky, nebo nakládky. V Transhipment problému ovšem nebyl definován depot a jak jsme uvedli výše, není to typický rozvozní problém. Naproti tomu v problému, která se v literatuře označuje jako PDP with Backhauls (Gribkovskaia a Laporte, 2008), se jedná o úlohu s depotem a vytvořenými trasami. V takové úloze jsou dva typy zákazníků (podobně jako v Transhipment problému), označovaní termíny *linehaul* (požadavek na doručení) a *backhaul* (požadavek na vyzvednutí).

Zmíněný problém se také někdy nazývá Clustered Travelling Salesman Problem (Chisman, 1975 in Gribkovskaia a Laporte, 2008). Množina všech uzlů U je tvořena třemi podmnožinami (clusters): $\{1\}$, $D = \{i \in U : d_i > 0 \text{ a } p_i = 0\}$ a $P = \{i \in U : p_i > 0 \text{ a } d_i = 0\}$, kde d_i je velikost požadavku na doručení a p_i je velikost požadavku na vyzvednutí. Tedy množina D obsahuje „linehaul“ uzly a množina P „backhaul“ uzly. Dostáváme se ke kombinovanému PDP, v němž je každý uzel i charakterizovaný dvěma hodnotami: $d_i \geq 0$ a $p_i \geq 0$. Může se tedy stát, že se v některých uzlech zároveň doručuje i vyzvedává, přičemž jde samozřejmě o nehomogenní zboží, neboť nemá žádný smysl vyzvedávat zboží stejného typu jaké je doručováno. Následující úloha je rozšířením tohoto typu úloh, neboť je přesně určeno, které zboží (předměty) a v jakém množství mají být v určitém místě vyzvednuty a do jiných míst doručeny. Psaraftis (2011) uvádí podobnou úlohu, v níž je využito jedno a dvě vozidla.

Nechť n je počet regionálních středisek, která odpovídají uzlům distribuční sítě. Jsou známé nejkratší vzdálenosti c_{ij} $(i, j = 1,2,...,n)$. Pro každou dvojici míst k a l je zadána hodnota q_{kl} odpovídající množství zboží, které je nutné přepravit z místa k do místa l. K dispozici jsou vozidla s kapacitou V. Na rozdíl od úloh uváděných v literatuře není depot pevně daný, ale může jím být jakýkoli uzel distribuční sítě. Požadavek q_{kl} může být rozdělen do několika částí (a to i v případě, že $q_{kl} \leq V$), které jsou pak přepravovány na různých trasách. Zboží může být dokonce přeloženo z jednoho vozidla na jiné v uzlu, který je společný trasám obou vozidel. Pro tuto úlohu lze formulovat matematický model vycházející z víceproduktové tokové úlohy (Pelikán a Fábry, 2012).

Do distribuční sítě přidáme dva fiktivní uzly, které označíme 0 a $n + 1$. V tokové úloze se jedná o vstup (zdroj) a výstup (místo určení). Zavedeme dva typy proměnných:

$y_{ij} \in Z_0^+$, celočíselná proměnná, která odpovídá počtu vozidel jedoucích po hraně (i, j) z uzlu i do uzlu j $(i, j = 0,1,...,n+1, i \neq j)$,

$x_{ij}^{kl} \in R_0^+$, množství zboží (část celkového požadavku q_{kl}) přepravovaného u uzlu i do uzlu j $(i, j = 0,1,...,n+1, i \neq j;\ k,l = 1,2,...,n, k \neq l)$.

Matematický model lze pak formulovat následujícím způsobem:

$$\text{minimalizovat} \quad z = \sum_{\substack{i=1 \\ i \neq j}}^{n} \sum_{j=1}^{n} c_{ij} y_{ij}, \tag{4.68}$$

za podmínek

$$\sum_{\substack{i=1 \\ i \neq j}}^{n} y_{ij} = \sum_{\substack{i=1 \\ i \neq j}}^{n} y_{ji}, \quad j = 1,2,...,n, \ , \tag{4.69}$$

$$\sum_{\substack{i=0 \\ i \neq j}}^{n} x_{ij}^{kl} = \sum_{\substack{i=1 \\ i \neq j}}^{n+1} x_{ji}^{kl}, \quad j = 1,2,...,n; k,l = 1,2,...n, k \neq l, \tag{4.70}$$

$$\sum_{k=1}^{n} \sum_{\substack{l=1 \\ k \neq l}}^{n} x_{ij}^{kl} \leq V y_{ij}, \quad i, j = 1,2,...,n, i \neq j, \tag{4.71}$$

$$x_{0,k}^{kl} = q_{kl}, \quad k,l = 1,2,...,n, k \neq l \ , \tag{4.72}$$

$$x_{0,j}^{kl} = 0, \quad k,l = 1,2,...,n, k \neq l; \ j = 1,2,...,n+1, \ j \neq k, \tag{4.73}$$

$$x_{l,n+1}^{kl} = q_{kl}, \quad k,l = 1,2,...,n, k \neq l, \tag{4.74}$$

$$x_{j,n+1}^{kl} = 0, \quad k,l = 1,2,...,n, k \neq l; \ j = 0,1,...,n, \ j \neq l, \tag{4.75}$$

$$x_{ij}^{kl} \in R_0^+, \quad i, j = 0,1,...,n+1, i \neq j; k,l = 1,2,...,n, k \neq l, \tag{4.76}$$

$$y_{ij} \in Z_0^+, \quad i, j = 1,2,...,n, i \neq j. \tag{4.77}$$

Účelová funkce (4.68) představuje celkovou vzdálenost, kterou všechna vozidla ujedou, tj. celkovou délku všech tras. Rovnice (4.69) zajišťují, že z každého místa odjede tolik vozidel, kolik jich do něj přijede. Podmínky (4.70) garantují, že množství zboží přepravované mezi místy k a l, které bude do určitého místa přivezeno, bude z tohoto místa také odvezeno. Bilanci mezi celkovým přepravovaným množstvím po hraně (i, j) a celkovou kapacitou všech vozidel, na tuto přepravu použitých, definují rovnice (4.71). Podmínky (4.72) – (4.75) zajistí, že

mezi uzly *k* a *l* bude přepraveno požadované množství q_{kl}, které odpovídá toku z uzlu 0 do uzlu *k* a zároveň z uzlu *l* do uzlu *n* + 1. Hodnota toku z 0 do jiných uzlů než *k* a do uzlu *n* + 1 z jiných uzlů než *l* je nulová.

Optimální hodnoty proměnných y_{ij} určují počty vozidel, které budou použity pro přepravu zboží po hraně (i, j), nelze z nich však získat informaci o tom, odkud tato vozidla vyjíždějí. Na rozdíl od úloh podobného typu totiž, jak bylo uvedeno výše, není předem definovaný depot, resp. depoty. Proto je nutné následně vygenerovat, pro nalezené optimální řešení, jednotlivé trasy. K tomu je možné použít tento postup vytváření cyklických tras v podobě cesty $(i_1, i_2, ..., i_t)$ (Pelikán a Fábry, 2012):

Krok 1: {Nalezení výchozí hrany (i_1, i_2), kde i_1 je depotem pro generovanou trasu}

Jestliže je $y_{ij} = 0$ pro všechny hrany, pak není možné generovat žádnou trasu, jinak vybereme jakoukoli hranu (i_1, i_2), pro kterou je $y_{i_1,i_2} > 0$, a následně nastavíme $y_{i_1,i_2} = y_{i_1,i_2} - 1$ a $t = 2$.

Krok 2: {Zařazení další hrany (i_t, i_{t+1}) do trasy}

Opakujeme dokud $i_t \neq i_1$: vybereme jakoukoli hranu (i_t, i_{t+1}), pro kterou je $y_{i_t,i_{t+1}} > 0$, a následně nastavíme $y_{i_t,i_{t+1}} = y_{i_t,i_{t+1}} - 1$ a $t = t + 1$.

Věta: Jestliže $\sum_{\substack{i=1 \\ i \neq j}}^{n} y_{ij} = \sum_{\substack{i=1 \\ i \neq j}}^{n} y_{ji}$ pro $j = 1,2,...,n$, pak předchozí algoritmus generuje cykly.

Důkaz: Pokud by v Kroku 2 cesta $(i_1, i_2, ..., i_t)$ nebyla uzavřená, tj. $y_{i_t,j} = 0$ pro všechny uzly *j*, pak $1 < \sum_{\substack{i=1 \\ i \neq j}}^{n} y_{ij} \neq \sum_{\substack{i=1 \\ i \neq j}}^{n} y_{ji} = 0$, což je v rozporu s předpokladem.

Na výše uvedeném postupu je založen heuristický algoritmus pro generování tras s minimalizací počtu přeložení nákladů mezi vozidly. Namísto libovolného výběru, budou hrany vybírány na základě určitého pravidla. Algoritmus vychází z optimálního řešení matematického modelu (4.68) – (4.77).

Základní myšlenkou heuristické metody je přepravit zboží z uzlu *k* do uzlu *l*, pokud možno bez přeložení na jiná vozidla, tj. v rámci jedné trasy (jedním vozidlem). Protože je prakticky nemožné všechny požadavky přepravit tímto způsobem, budeme na začátku pro přepravu

preferovat velké objemy zboží. V následujícím popisu označíme *E* množinu všech hran a *U* množinu všech uzlů.

Krok 1: Jestliže je $y_{ij} = 0$ pro všechny hrany, jdi na Konec.

Krok 2: {Nalezení výchozí hrany $(i_1, i_2) \in E$, kde $i_1 \in U$ je depotem pro generovanou trasu}

i) Najdi maximální hodnotu přepravy (toku) požadavku q_{vw} z uzlu *v* do uzlu *w* po hraně (v, j^*), na níž je k dispozici alespoň jedno vozidlo:

$$x_{vj^*}^{vw} = \max_{\substack{k,l \in U \\ (k,j) \in E | y_{kj} > 0}} x_{kj}^{kl}, \tag{4.78}$$

a nastav $i_1 = v$; $i_2 = j^*$.

ii) Pokud maximum (4.78) neexistuje, najdi maximální hodnotu (tok) požadavku q_{vw} z uzlu *v* do uzlu *w* po hraně (i^*, j^*) podle:

$$x_{i^* j^*}^{vw} = \max_{\substack{k,l \in U \\ (i,j) \in E | y_{ij} > 0}} x_{ij}^{kl}, \tag{4.79}$$

a nastav $i_1 = i^*$; $i_2 = j^*$.

Nastav $y_{i_1,i_2} = y_{i_1,i_2} - 1$; $r = 1$; $t = 2$.

Krok 3: {Zařazení další hrany (i_t, i_{t+1}) do trasy}

Opakuj, dokud $i_t \neq i_1$:

3a) {Zboží z uzlu *v* do uzlu *w* je přepravováno po hraně (i_t, i_{t+1}) bez přeložení}

Jestliže $w \neq 0$, pak

dokud $i_t \neq w$ opakuj

$$x_{i_t j^*}^{vw} = \max_{(i_t, j) \in E | y_{i_t j} > 0} x_{i_t j}^{vw}, \tag{4.80}$$

nastav $i_{t+1} = j^*$; $y_{i_t,i_{t+1}} = y_{i_t,i_{t+1}} - 1$; $t = t + 1$.

3b) {Zboží z uzlu *v* do uzlu *w* je v uzlu $i_t = w$ vyloženo a je hledán další tok}

Jestliže $i_t = w$, pak vypočti

$$d = \min_{s=r,r+1,\ldots,t-1} x_{i_s i_{s+1}}^{vw}, \tag{4.81}$$

a proveď změnu:

$$x_{i_s i_{s+1}}^{vw} = x_{i_s i_{s+1}}^{vw} - d, \quad s = r, r+1, ..., t-1. \tag{4.82}$$

Pokud $\exists l \in U$ a $(i_t, j) \in E \mid x_{i_t j}^{i_t l} > 0$, najdi tok (část požadavku $q_{i_t w}$) z uzlu i_t (nový uzel *v*) do nového uzlu *w* po hraně (i_t, j^*):

$$x_{vj^*}^{vw} = \max_{\substack{l \in U \\ (i_t, j) \in E \mid y_{i_t j} > 0}} x_{i_t j}^{i_t l}, \tag{4.83}$$

a nastav $i_{t+1} = j^*$; $y_{i_t, i_{t+1}} = y_{i_t, i_{t+1}} - 1$; $r = t$; $t = t+1$

jinak

pokud $\exists k, l \in U$ a $(i_t, j) \in E \mid x_{i_t j}^{kl} > 0$, najdi tok (část požadavku q_{vw}) z nového uzlu $v \neq i_t$ do nového uzlu *w* po hraně (i_t, j^*):

$$x_{i_t j^*}^{vw} = \max_{\substack{k,l \in U \\ (i_t, j) \in E \mid y_{i_t j} > 0}} x_{i_t j}^{kl}, \tag{4.84}$$

a nastav $i_{t+1} = j^*$; $y_{i_t, i_{t+1}} = y_{i_t, i_{t+1}} - 1$; $r = t$; $t = t+1$

jinak

nastav $w = 0$ a vyber hranu $(i_t, i_{t+1}) \in E \mid y_{i_t i_{t+1}} > 0$;

nastav $y_{i_t, i_{t+1}} = y_{i_t, i_{t+1}} - 1$; $t = t+1$.

Jdi na Krok 3.

Krok 4: {Trasa je ukončena v uzlu i_1}

Jestliže $w \neq 0$ pak podle (4.81) a (4.82) proveď změnu toků po hranách trasy.

Uvedený algoritmus najde jednu trasu s předem neurčeným depotem. Jeho opakováním vytvoříme trasy bez překládání zboží pro všechna vozidla určená optimalizačním modelem (4.68) – (4.77). Dodatečně je nutné zbývající množství rozdělit na nenaplněná vozidla tak, aby bylo veškeré zboží přepraveno. V práci Bartáskové (2011) jsou model a výše uvedený algoritmus aplikovány na reálnou úlohu nadnárodní dopravní společnosti, zabývající se mj. pozemní přepravou větších zásilek mezi distribučními centry. V tomto případě se jedná o 9 velkých měst v České republice a přeprava mezi nimi se uskutečňuje kamiony.

Existuje celá řada dalších problémů typu PDP. Jejich prezentace v této práci je však nad její rámec.

5 Arc Routing

Jak bylo výše uvedeno, česká terminologie v oblasti okružních a rozvozních úloh není příliš bohatá a mnohdy přesně nevystihuje podstatu daného problému. Totéž platí o úlohách, jimž je věnována tato kapitola, a které se v anglické terminologii zahrnují pod pojem Arc Routing Problems (ARP). V českém jazyce se se dosud nevžil žádný vhodný ekvivalent, a proto se problémy tohoto typu přiřazují k původní úloze (čínského) listonoše CPP[65], kterou takto pojmenoval poprvé Edmonds (1965) na počest čínského matematika Kwana, též zvaného Guana (1962), který tento problém jako první definoval. V původní verzi se jedná o úlohu, v níž je v neorientovaném grafu definována množina ohodnocených hran, které je všechny nutné v cyklu projet (projít) alespoň jednou, tedy najít takový uzavřený sled, který obsahuje každou hranu alespoň jednou. Pokud je graf eulerovský (všechny uzly jsou sudého stupně), je každá hrana vybrána právě jednou. V opačném případě je nutné některé hrany zdvojit tak, aby celková délka výsledného sledu byla minimální.

Oblast ARP je širší skupinou úloh, jejíž hlavní částí jsou ovšem nejrůznější modifikace CPP. Protože označení *čínský listonoš* má jen historické důvody, budeme v dalším textu a v konkrétních modifikacích (Eiselt et al., 1995a a 1995b; Assad a Golden, 1995) uvádět název *problém listonoše*.

Pro následující klasifikaci úloh označme graf $G = \{U, H\}$, U množinu všech uzlů, H množinu všech hran, E množinu všech neorientovaných hran, A množinu všech orientovaných hran a R množinu všech povinných hran (tj. hran, které je nutné projít či projet). V neorientovaném problému listonoše UPP[66] předpokládáme všechny hrany neorientované, tedy $R = E = H$. Pro jeho řešení lze použít polynomiální algoritmus (např. Edmonds a Johnson, 1973) pro párování (matching). V orientovaném problému listonoše DPP[67] jsou naopak všechny hrany orientované, tedy $R = A = H$, a k jeho řešení lze využít polynomiální algoritmy pro řešení tokových úloh (Edmonds a Johnson, 1973, Orloff, 1974 aj.). Smíšený problém listonoše MPP[68], který obsahuje minimálně jednu hranu neorientovanou a alespoň jednu hranu orientovanou, tedy $R = H = E \cup A$, $E \neq \emptyset$, $A \neq \emptyset$, patří mezi NP-obtížné problémy (Papadimitriou, 1976). Následující NP-obtížné modifikace CPP nemají v českém jazyce odpovídající ekvivalent, proto budou uvedeny v původním anglickém označení.

[65] Chinese Postman Problem.
[66] Undirected Postman Problem.
[67] Directed Postman Problem.
[68] Mixed Postman Problem.

Rural Postman Problem (RPP) je velice zajímavým rozšířením CPP (Orloff, 1974, Lenstra a Rinooy Kan, 1976, Eiselt et al., 1995b aj.). Hrany se rozdělují na povinné a nepovinné. Zatímco povinné hrany, jejichž množinu označíme $R \subset H$, je nutné projít (projet), nepovinné hrany mohou, ale nemusí být využity k získání efektivního řešení. Také v této speciální definici úlohy se může jednat o neorientovanou a orientovanou verzi grafu, tedy URPP a DRPP[69]. V případě URPP je $R \subset E = H$, v úloze DRPP pak $R \subset A = H$. Zvláštní název má úloha RPP na smíšeném grafu (zvláštní případ MPP), v níž jsou povinné všechny orientované hrany, tedy $R = A$, zatímco neorientované hrany jsou nepovinné. Tato úloha, jejíž řešení uvádějí Eiselt et al. (1995b), se nazývá Stacker Crane Problem.

Další speciální úlohou je tzv. Windy Postman Problem (WPP), kterou uvedl Minieka (1979). Jedná se sice o úlohu na neorientovaném grafu, kde $R = E$, ale ohodnocení hran může být závislé na směru, v němž je hrana projeta. Otázkou řešitelnosti WPP se zabývá Guan (1984). Zvláštní varianta této úlohy je Windy Rural Postman Problem (Benavant et al., 2005), v níž je $R \subset E$.

Pokud je v úloze CPP zadána preferenční relace na množině hran *H*, jedná se o tzv. hierarchický problém listonoše HPP[70] (Dror et al., 1987, Eiselt et al., 1995a). V tomto případě je nutné některé hrany obsloužit dříve než jiné hrany, případně respektovat pěvně zadanou posloupnost obsloužených hran. Nejznámější reálnou aplikací je úklid sněhu či posyp komunikací, kdy jsou jednotlivé silnice či ulice rozděleny do několika tříd podle jejich důležitosti. Exaktní algoritmus, který nabízejí Ghiani a Improta (2000), předpokládá rozdělení hran do *p* tříd $H_1, H_2, ..., H_p$ takových, že $H_1 \cup H_2 \cup ... \cup H_p$ a $H_i \cap H_j = \emptyset \ \forall i, j = 1,2,..., p, i \neq j$, přičemž žádná hrana z množiny H_i nesmí být obsloužena dříve, než jsou obslouženy všechny hrany z množiny H_{i-1} $(i = 2,3,..., p)$. S tímto předpokladem pracují i Cabral et al. (2004), kteří úlohu HPP transformují na RPP.

Kapacitní úloha čínského listonoše CAPP[71] je analogií (kapacitních) rozvozních úloh. Na rozdíl od nekapacitních verzí úloh listonoše je nutné uvažovat kapacitu obsluhujícího vozidla, a to vzhledem k nenulovým požadavkům obsluhovaných hran. Golden a Wong (1981, in Assad a Golden, 1995) formulovali matematický model celočíselného programování pro orientovaný graf, Belenguer a Benavant (1992, in Eiselt et al., 1995b) pro graf neorientovaný.

[69] Undireceted Rural Postman Problem a Direceted Rural Postman Problem.
[70] Hierarchical Postman Problem.
[71] Capacitated Postman Problem.

Podrobněji se tomuto tématu bude věnovat samostatná podkapitola. Pro následující text je důležitý pojem unicursalita (Ford a Fulkerson, 1962 in Eiselt et al., 1995b). Graf je unicursální, pokud:

a) v neorientovaném grafu jsou všechny uzly sudého stupně,

b) v orientovaném grafu je pro každý uzel jeho vnitřní polostupeň roven vnějšímu polostupni,

c) ve smíšeném grafu musí být každý uzel incidentní se sudým počtem orientovaných a neorientovaných hran; navíc pro každou podmnožinu $S \subseteq U$ musí platit, že rozdíl mezi počtem orientovaných hran směřujících z S do $U - S$ a počtem orientovaných hran směřujících z $U - S$ do S musí být menší nebo roven počtu neorientovaných hran spojujících S a $U - S$.

5.1 Neorientovaný problém listonoše

V úloze UPP předpokládáme hranově ohodnocený neorientovaný graf $G = \{U, E\}$, v němž je nutné najít cyklus, který bude obsahovat každou hranu alespoň jednou a přitom součet ohodnocení hran v tomto cyklu bude minimální. Následující matematický model (Pelikán, 2001) je úpravou modelu Edmondse a Johnsona (1973). Nechť $U = \{1,2,...,n\}$. Vzhledem k tomu, že se jedná o neorientovaný graf, definujeme pro každou dvojici uzlů i a j hodnotu $c_{ij} = c_{ji}\ (i, j = 1,2,...,n)$. Pro existující hrany $(i, j) \in E$ se jedná o její délku, v případě, že $(i, j) \notin E$, dosadíme za tyto hodnoty velkou konstantu M. Protože se jedná o symetrickou matici, budou nás zajímat pouze hodnoty[72] c_{ij} pro $i < j$. Podobně lze definovat binární proměnné x_{ij} jen pro $i < j$, tedy pro $i = 1,2,...,n-1,\ j = i+1, i+2,...,n$. Pokud je hrana $(i, j) \in E$, zahrnuta do grafu ještě jednou (zdvojena) za účelem získání eulerovského grafu, pak $x_{ij} = 1$, v opačném případě $x_{ij} = 0$.

Matematický model pak zapíšeme v následujícím tvaru:

$$\text{minimalizovat} \quad z = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} x_{ij}\ , \tag{5.1}$$

[72] Z tohoto důvodu nemusíme řešit hodnoty na diagonále, tedy c_{ii}.

za podmínek

$$\sum_{j=1}^{i-1} x_{ji} + \sum_{j=i+1}^{n} x_{ij} = 2y_i, \quad i \in U_0 \ , \tag{5.2}$$

$$\sum_{j=1}^{i-1} x_{ji} + \sum_{j=i+1}^{n} x_{ij} = 2y_i + 1, \quad i \in U_1 \ , \tag{5.3}$$

$$x_{ij} \in \{0,1\}, \quad i = 1,2,...,n-1, \quad j = i+1, i+2,...,n, \tag{5.4}$$

$$y_i \in Z_0^+, \quad i = 1,2,...,n \ . \tag{5.5}$$

Účelová funkce (5.1) představuje součet ohodnocení přidaných hran, tedy celkovou vzdálenost, kterou vozidlo ujede navíc oproti vzdálenosti, kterou vozidlo musí ujet, aby obsloužilo všechny hrany. Množina uzlů U je rozdělena na dvě disjunktní podmnožiny U_0 a U_1, z nichž první představuje množinu uzlů sudého stupně a druhá množinu uzlů lichého stupě. Jak vyplývá ze známé nutné podmínky existence Eulerova cyklu, musí mít všechny uzly sudý stupeň. Pokud tedy v původním grafu některé hrany zdvojujeme, musíme k uzlu s lichým stupněm přidat lichý počet hran, k uzlu se sudým stupněm sudý počet hran, případně k němu žádné hrany nepřidáváme. To zajišťují celočíselné proměnné y_i v podmínkách (5.2) a (5.3).

Na základě řešení výše uvedeného modelu je pak nutné najít Eulerův cyklus některým ze známých postupů (např. pomocí Fleuryho algoritmu).

5.2 Orientovaný problém listonoše

V úloze DPP je dán hranově ohodnocený orientovaný graf $G = \{U, A\}$. Pokud má vozidlo projet každou orientovanou hranou alespoň jednou, je nutné, aby v rozšířeném grafu pro každý uzel platilo, že počet orientovaných hran vedoucích do tohoto uzlu bude roven počtu orientovaných hran z něj vedoucích. Toto je podmínka nutná a postačující pro existenci Eulerova cyklu. Označíme-li $\deg_i^-$ vnitřní polostupeň uzlu[73] i, tj. počet hran vstupujících do uzlu i a $\deg_i^+$ vnější polostupeň uzlu[74] i, tj. počet hran vystupujících z uzlu i, pak existence Eulerova cyklu vyžaduje rovnost $\deg_i^- = \deg_i^+ \ \forall i \in U$. Assad a Golden (1995) pro získání optimálního řešení uvádějí algoritmus, který vychází z přístupu Edmondse a Johnsona (1973), a jehož mírně upravená verze je uvedena v následujícím textu.

[73] In-degree.
[74] Out-degree.

Opět předpokládáme ohodnocení (délku) hran $c_{ij}\ \forall(i,j)\in A$. Označme $b_i = \deg_i^- - \deg_i^+$ a definujme dvě podmnožiny uzlů $I, J \subset U$:

$$I = \{ i \in U; b_i > 0 \} \text{ a } J = \{ j \in U; b_j < 0 \}. \tag{5.6}$$

Pro každou dvojici uzlů $i \in I$, $j \in J$ vypočteme délku p_{ij} nejkratší cesty P_{ij} z uzlu *i* do uzlu *j*. Lze tedy psát:

$$p_{ij} = \sum_{(v,w)\in P_{ij}} c_{vw}\,. \tag{5.7}$$

Řešíme dopravní problém, v němž je dáno $|I|$ dodavatelů s kapacitami b_i pro $i \in I$ a $|J|$ odběratelů s požadavky $|b_j|$ pro $j \in J$. Jednotkové přepravní náklady jsou definovány hodnotami (5.7). Pro každou dvojici uzlů $i \in I$, $j \in J$ zavedeme proměnnou $y_{ij} \in Z_0^+$, jejíž optimální hodnota představuje počet nejkratších orientovaných cest p_{ij} mezi uzly *i* a *j*, které jsou přidané k původnímu grafu. Minimalizační účelová funkce

$$\sum_{i\in I}\sum_{j\in J} p_{ij} y_{ij} \tag{5.8}$$

představuje celkovou vzdálenost, kterou vozidlo ujede navíc oproti vzdálenosti, kterou vozidlo musí ujet, aby obsloužilo všechny hrany. Jestliže označíme $x_{ij} \in Z_0^+$ počet „kopií“ hrany $(i,j) \in A$, pak její hodnotu lze určit takto:

$$x_{ij} = \sum_{(i,j)\in P_{vw}} y_{vw}\,. \tag{5.9}$$

Výraz na pravé straně rovnice (5.9) představuje celkový počet cest (z řešení výše uvedeného dopravního problému), jejichž součástí je hrana (i,j). Přidáme-li hrany na základě (5.9), získáme eulerovský graf.

5.3 Kapacitní problém listonoše

Úloha CAPP je definována pro neorientované i orientované grafy (Longo et al., 2006). Tato práce se soustředí pouze na neorientované grafy, navíc půjde o rozšíření úlohy URPP. Eiselt et al. (1995b) nabízejí „orientované“ řešení úlohy, v němž jsou definovány proměnné, definující směr průjezdu každou hranou. K řešení úlohy lze použít i „neorientovanou“ variantu (Fábry a Pelikán, 2013), pro rozsáhlejší úlohy je nutné použít některou z heuristických metod.

5.3.1 Orientované řešení pro neorientovaný graf

Předpokládáme hranově ohodnocený neorientovaný graf $G=\{U,E\}$ a množinu povinných hran $R\subseteq E$. Pro všechny hrany $(i,j)\in E$ jsou definovány jejich délky $c_{ij}=c_{ji}$, pro hrany $(i,j)\in R$ jsou navíc definovány požadavky na svoz (rozvoz) $d_{ij}>0$. Uzel č. 1 představuje depot, v němž je umístěno *K* vozidel o kapacitě *V*. Přestože je graf neorientovaný, řešení problému bude určovat směr průjezdu hrany. Cílem je tedy najít orientované cyklické trasy.

Zavedeme binární proměnnou x_{ij}^k, která nabývá hodnoty 1, jestliže *k*-té vozidlo projíždí hranou $(i,j)\in E$ ve směru z uzlu *i* do uzlu *j*, hodnoty 0 v opačném případě. Další binární proměnná y_{ij}^k je rovna 1 v případě, že hrana $(i,j)\in R$ je obsloužena vozidlem na *k*-té trase, tj. požadavek d_{ij} je naložen (při svozu) na *k*-té vozidlo, jinak je proměnná rovna 0. Matematický model lze pak formulovat následovně (Eiselt et al., 1995b):

$$\text{minimalizovat}\quad z=\sum_{k=1}^{K}\sum_{(i,j)\in E}c_{ij}x_{ij}^k, \tag{5.10}$$

za podmínek

$$y_{ij}^k\le x_{ij}^k,\quad \forall(i,j)\in E,\quad k=1,2,...,K, \tag{5.11}$$

$$\sum_{k=1}^{K}\left(y_{ij}^k+y_{ji}^k\right)=1,\quad \forall(i,j)\in R, \tag{5.12}$$

$$\sum_{(i,j)\in R}d_{ij}y_{ij}^k\le V,\quad k=1,2,...,K, \tag{5.13}$$

$$\sum_{(i,j)\in E}x_{ij}^k=\sum_{(j,i)\in E}x_{ji}^k,\quad \forall j\in U,\quad k=1,2,...,K, \tag{5.14}$$

$$\sum_{\substack{(i,j)\in E\\ i,j\in S}}\left(x_{ij}^k+x_{ji}^k\right)\le|S|-1+|U|^2u_k^S,\quad k=1,2,...,K,\quad S\subseteq U-\{1\},S\ne\emptyset, \tag{5.15}$$

$$\sum_{\substack{i\in S\\ (i,j)\in E}}\sum_{j\notin S}x_{ij}^k\ge 1-w_k^S,\quad k=1,2,...,K,\quad S\subseteq U-\{1\},S\ne\emptyset, \tag{5.16}$$

$$u_k^S+w_k^S\le 1,\quad k=1,2,...,K,\quad S\subseteq U-\{1\},S\ne\emptyset, \tag{5.17}$$

$$x_{ij}^k\in\{0,1\},\ y_{ij}^k\in\{0,1\},\quad \forall(i,j)\in E,\quad k=1,2,...,K, \tag{5.18}$$

$$u_k^S\in\{0,1\},\ w_k^S\in\{0,1\},\quad k=1,2,...,K,\quad S\subseteq U-\{1\},S\ne\emptyset. \tag{5.19}$$

Účelová funkce (5.10) představuje celkovou vzdálenost, kterou ujedou všechna vozidla, tj. celkovou délku všech tras. Nerovnosti (5.11) zajišťují, aby v případě, že hrana $(i, j) \in R$ je obsloužena na *k*-té trase ($y_{ij}^k = 1$), bude součástí této trasy ($x_{ij}^k = 1$). Pokud požadavek d_{ij} na hraně $(i, j) \in R$ je obsloužen na *k*-té trase, pak buď $y_{ij}^k = 1$ nebo $y_{ji}^k = 1$. Protože každá povinná hrana musí být obsloužena právě jednou, musí být splněny podmínky (5.12). Nerovnosti (5.13) zabrání překročení kapacity vozidel.

Důležitou podmínkou pro existenci cyklických tras je výše zmíněná unicursalita neorientovaného grafu, tj. vzhledem k definici proměnných x_{ij}^k, musí platit, že počet „vjezdů“ do každého uzlu musí být roven počtu „výjezdů“, což zajišťují rovnice (5.14). Tyto podmínky ovšem nezaručují souvislost tras. Proto je nutné do modelu zahrnout podmínky (5.15) – (5.17), které zabraňují vzniku zakázaných cyklů, tj. takových tras, které neobsahují depot. Předpokládejme $G' = \{S, E(S)\}$, kde $S \subset U$ a $E(S)$ je množina hran spojující tyto uzly[75]. Pokud je počet hran $E(S)$ menší než počet uzlů S, tj. platí $|E(S)| < |S|$, pak G' nepředstavuje cyklickou trasu. Pokud ovšem $|E(S)| \geq |S|$ a navíc $S \subseteq U - \{1\}$, pak existuje možnost, že graf G' obsahuje zakázaný cyklus, a musí být přidána alespoň jedna hrana spojující množiny uzlů S a $U - S$. Počet hran v množině $E(S)$ lze zapsat jako

$$|E(S)| = \sum_{\substack{(i,j)\in E \\ i,j\in S}} \left(x_{ij}^k + x_{ji}^k\right), \quad k = 1,2,...,K\,. \tag{5.20}$$

Pokud tedy chceme zabránit vzniku zakázaných cyklů, je nutné zavést pro každou podmnožinu *S* a každé vozidlo *k* další binární proměnné u_k^S a w_k^S, s jejichž pomocí naformulujeme podmínky (5.15) – (5.17). Pokud $|E(S)| \geq |S|$, pak $u_k^S = 1$ a $w_k^S = 0$. Pak nerovnosti (5.16) zaručí, že každá množina S, která neobsahuje depot, bude spojena alespoň jednou hranou s množinou $U - S$, která obsahuje depot.

Existují dva způsoby, jak lze uvedený model aplikovat při řešení úlohy:

a) Definujeme všechny podmnožiny $S \subseteq U - \{1\}$ a pro ně příslušné podmínky (5.15) – (5.17). Takto vytvořený model pak řešíme. Nevýhodou tohoto přístupu je výpočetní náročnost vzhledem k enormně vysokému počtu množin *S*.

[75] Může se jednat o multigraf, který spojuje uzly množiny *S* a v němž jsou některé hrany zdvojené.

b) Z matematického modelu jsou vypuštěny podmínky eliminující zakázané trasy a takový model je pak vyřešen. Pokud vznikne zakázaný cyklus, jsou výše zmíněné podmínky zahrnuty pro množinu *S*, která obsahuje uzly, jež jsou součástí zakázaného cyklu. Takto rozšířený model je pak znovu řešen. Proces se opakuje, dokud není získáno přípustné řešení původní úlohy, tedy řešení bez zakázaných cyklů. Ačkoli samotný model není tak složitý (oproti plnému modelu vytvořenému pro všechny podmnožiny *S*, zvláště na počátku optimalizačního procesu), nevýhodou je opakované řešení modelu, který se postupně stává složitějším tím, jak se postupně přidávají další omezující podmínky.

5.3.2 Neorientované řešení pro neorientovaný graf

V předchozí části byl popsán matematický model, jehož řešením získáme orientované cyklické trasy obsahující depot. V této části se soustředíme na možnost řešení pomocí modelu, který generuje množiny hran, jež jsou součástí cyklické trasy, bez určení její orientace. Orientovaná trasa je nalezena až následně např. Fleuryho algoritmem. Zavedeme celočíselnou proměnnou x_{ij}^k, jejíž hodnota představuje počet průjezdů *k*-tého vozidla hranou $(i, j) \in E$ (v jakémkoli směru). Binární proměnná y_{ij}^k má stejný význam jako v předchozím modelu. Obě proměnné jsou navíc definovány, podobně jako u symetrické verze TSP, pouze pro $i < j$.

Zatímco účelová funkce (5.10) a omezující podmínky (5.11) a (5.13) zůstávají beze změny, podmínky (5.12) a (5.14) je potřeba upravit následovně:

$$\sum_{k=1}^{K} y_{ij}^k = 1, \quad \forall (i, j) \in R, \tag{5.12a}$$

$$\sum_{(i,j)\in E} x_{ij}^k + \sum_{(j,i)\in E} x_{ji}^k = 2 f_j^k, \quad \forall j \in U, \quad k = 1,2,...,K. \tag{5.14a}$$

Celočíselná proměnná f_j^k představuje počet průjezdů uzlu *j* *k*-tým vozidlem. Proměnné (5.19) jsou z modelu vypuštěny, stejně jako podmínky (5.15) a (5.17). Nerovnosti (5.16) jsou pak nahrazeny následujícími podmínkami se stejným významem:

$$\sum_{\substack{i\in S \\ (i,j)\in E}} \sum_{j\notin S} x_{ij}^k + \sum_{\substack{j\in S \\ (j,i)\in E}} \sum_{i\notin S} x_{ji}^k \geq \frac{1}{M} \sum_{\substack{i\in S \\ (i,j)\in E}} \sum_{j\in S} x_{ij}^k, \quad k = 1,2,...,K, \quad S \subseteq U - \{1\}, S \neq \emptyset, \tag{5.16a}$$

kde *M* je vysoká konstanta. Vzhledem k tomu, že kapacitní úloha listonoše patří mezi NP-obtížné problémy, při jejím větším rozměru nelze očekávat získání optimálního řešení, což bylo ověřeno v práci Bilé (2013). V takovém případě je nutné použít přibližné metody.

5.3.3 Heuristické algoritmy pro vytvoření cyklických tras

Nejprve popíšeme heuristický algoritmus, který pří vytváření tras využívá deterministický přístup k výběru hran s využitím nejkratších cest mezi uzly. Druhý přístup je založený na náhodném výběru hran. Stejně jako v předchozích modelech, i v následujícím textu budeme předpokládat existenci povinných a nepovinných hran.

5.3.3.1 Algoritmus založený na deterministickém výběru hran

Navržený algoritmus (Fábry a Pelikán, 2013) nalezne cyklické trasy dané posloupností uzlů $u_1^k, u_2^k, \ldots$ (k je číslo trasy, resp. vozidla, které tuto trasu absolvuje). Jak bylo uvedeno výše, metoda využívá nejkratší cesty mezi uzly. Označme p_{ij} délku nejkratší cesty mezi uzly i a j. Celková délka cyklické trasy k je označena F_k, celkový náklad na vozidle L_k a počet uzlů na trase N_k. R je množina všech dosud neobsloužených povinných hran, V je kapacita vozidla, d_{ij} je požadavek hrany $(i, j) \in R$ na svoz a c_{ij} je délka hrany $(i, j) \in E$.

Krok 1: $k = 0$.

Krok 2: Jestliže $R = \emptyset$, pak jdi na Krok 5

jinak

$k = k + 1;\ L_k = 0;\ F_k = 0;\ t = 1;\ u_t^k = 1;\ N_k = 1.$

Krok 3: Jestliže $R = \emptyset$, pak

$u_{t+1}^k = 1;\ N_k = N_k + 1;\ F_k = F_k + p_{u_t^k 1}$; jdi na Krok 5.

Krok 4: Jestliže $\exists (u_t^k, j) \in R \mid L_k + d_{u_t^k j} \le V$ pak

$F_k = F_k + c_{u_t^k j};\ L_k = L_k + d_{u_t^k j};\ N_k = N_k + 1;$

$R = R - \{(u_t^k, j)\};\ t = t + 1\ \ u_t^k = j;$

pokud $u_t^k = 1$ pak jdi na Krok 2 jinak jdi na Krok 3

jinak

jestliže $\exists (i, j) \in R, i \neq 1 \mid L_k + d_{ij} \le V$ pak

$F_k = F_k + p_{u_t^k i} + c_{ij};\ L_k = L_k + d_{ij};\ N_k = N_k + 2;$

$R = R - \{(i, j)\};\ u_{t+1}^k = i;\ u_{t+2}^k = j;\ t = t + 2;$

pokud $u_t^k = 1$ pak jdi na Krok 2 jinak jdi na Krok 3

jinak

$F_k = F_k + p_{u_t^k 1};\ N_k = N_k + 1;\ u_{t+1}^k = 1$; jdi na Krok 2.

Krok 5: $z = \sum_{l=1}^{k} F_l$.

Konec.

V Kroku 2 je v případě, že existuje ještě nějaká neobsloužená povinná hrana, inicializována nová trasa. Krok 3 uzavírá poslední trasu, pokud jsou všechny povinné hrany obslouženy. V Kroku 4 nejprve hledáme dosud neobslouženou hranu, která je incidentní s posledním uzlem na vytvářené trase a její požadavek lze uspokojit (vzhledem ke kapacitě vozidla). Pokud existuje takových hran několik, její výběr je náhodný. V případě, že taková hrana neexistuje, hledáme libovolnou povinnou hranu, jejíž požadavek lze uspokojit. Jestliže, vzhledem ke kapacitě vozidla *V*, ani taková hrana neexistuje, pak je nutné trasu ukončit návratem do depotu a zahájit novou trasu.

Výběr hran v Kroku 4 lze modifikovat. Je možné například vybrat hranu s největším požadavkem, který lze daným vozidlem uspokojit. V případě, že musí vozidlo dojet k některé povinné hraně, pak je vhodné vybrat hranu, která je vozidlu v daném kroku nejblíž. Pokud navíc existuje více povinných hran incidentních s tímto uzlem, opět vybereme hranu s největším požadavkem. Zápis této úpravy Kroku 4 je pak následující:

Krok 4: Jestliže $\exists(u_t^k, j) \in R \mid L_k + d_{u_t^k j} \le V$ pak

$d_{u_t^k j^*} = \max_{\substack{(u_t^k, j)\in R \\ L_k + d_{u_t^k j} \le V}} d_{u_t^k j}$;

$F_k = F_k + c_{u_t^k j^*}$; $L_k = L_k + d_{u_t^k j^*}$; $N_k = N_k + 1$;

$R = R - \{(u_t^k, j^*)\}$; $t = t + 1$ $u_t^k = j^*$;

pokud $u_t^k = 1$ pak jdi na Krok 2 jinak jdi na Krok 3

jinak

jestliže $\exists(i, j) \in R, i \ne 1 \mid L_k + d_{ij} \le V$ pak

$p_{u_t^k i^*} = \min_{\substack{(i,j)\in R \\ L_k + d_{ij} \le V}} p_{u_t^k i}$;

$d_{i^* j^*} = \max_{\substack{(i^*, j)\in R \\ L_k + d_{i^* j} \le V}} d_{i^* j}$;

$F_k = F_k + p_{u_t^k i^*} + c_{i^* j^*}$; $L_k = L_k + d_{i^* j^*}$; $N_k = N_k + 2$;

$R = R - \{(i^*, j^*)\}$; $u_{t+1}^k = i^*$; $u_{t+2}^k = j^*$; $t = t + 2$;

pokud $u_t^k = 1$ pak jdi na Krok 2 jinak jdi na Krok 3

jinak

$F_k = F_k + p_{u_t^k 1}$; $N_k = N_k + 1$; $u_{t+1}^k = 1$; jdi na Krok 2.

Ve výše uvedené metodě i v její modifikaci je pevně určen výběr hran. I v případě dalších modifikací dostáváme trasy získané deterministickým přístupem. V případě této úlohy se na-

bízí stochastický přístup k výběru hran, na kterém jsou založeny některé metaheuristické postupy. Jednoduchá randomizovaná heuristika je alternativním přístupem k řešení kapacitní úlohy listonoše.

5.3.3.2 Algoritmus založený na náhodném výběru hran

Metoda spočívá v opakovaném generování tras, kdy jsou hrany do trasy zařazovány na základě jejich náhodného výběru. Počet vygenerovaných řešení *Nmax* je předem zvolený a je použitý pro definování ukončovacího pravidla simulačního běhu. Během vytváření trasy se generuje posloupnost navštívených uzlů $U_k = \{u_1^k, u_2^k, ..., u_{h_k}^k\}$, kde $u_1^k = u_{h_k}^k = 1$ (1 je výchozí uzel), index *k* označuje pořadové číslo simulačního kroku, během něhož získáme jedno vygenerované řešení, a h_k je počet uzlů v posloupnosti U_k. Nechť *P* je množina nejkratších cest mezi uzly $i, j \in U$, jejichž délky označíme p_{ij}. Pokud hodnota F_k představuje délku vytvořené trasy, pak pro délku nejkratší dosažené trasy platí

$$F^* = \min_{k=1,2,...,Nmax} F_k . \tag{5.21}$$

Nejlepší trasu označíme U^*. Hodnota L_t představuje hodnotu nákladu na vozidle po zařazení uzlu u_t^k do posloupnosti U_k. Pro každou povinnou hranu $(i, j) \in R$ je definovaná binární proměnná $serv_{ij}$, jejíž hodnota je rovna 0 v případě, že hrana v generované trase dosud nebyla obsloužena, jinak nabývá hodnoty 1. Cílem je tedy dosáhnout rovnosti:

$$\sum_{(i,j)\in R} serv_{ij} = |R|, \tag{5.22}$$

kde $|R|$ je počet povinných hran. Navržený algoritmus je pak možné zapsat následovně:

Krok 1: $k = 1;\ F^* = +\infty$.

Krok 2: {zahájení trasy U_k}

Jestliže $k > Nmax$ jdi na Krok 9

$t = 1;\ u_1^k = 1;\ F_k = 0;\ L_t = 0;\ n_R = |R|;\ n_1 = n_2 = 0;\ serv_{ij} = 0\ \forall (i, j) \in R$.

Krok 3: {vytvoření množiny $R_t \subset R$ obsahující hrany incidentní s uzlem u_t^k}

Jestliže $n_R = 0$ pak jdi na Krok 7

$R_t = \{(u_t^k, j) \in R \mid serv_{u_t^k j} = 0, L_t + d_{u_t^k j} \le V\};\ n_1 = |R_t|;$

pokud $n_1 = 0$ jdi na Krok 5.

Krok 4: {výběr hrany z množiny R_t }

Vyber hranu (u_t^k, j^*) z množiny R_t náhodným způsobem

$u_{t+1}^k = j^*;\ F_k = F_k + c_{u_t^k u_{t+1}^k};\ L_{t+1} = L_{t+1} + d_{u_t^k u_{t+1}^k};\ t = t+1;$

$n_R = n_R - 1$; jestliže $n_R = 0$ jdi na Krok 7;

pokud $F_k \geq F^*$ pak

$k = k+1$; jdi na Krok 2;

jestliže $u_t^k = 1$ nastav $L_t = 0$;

jdi na Krok 3.

Krok 5: {vytvoření množiny $P_t \subset P$, obsahující všechny cesty z uzlu u_t^k k neobslouženým povinným hranám}

$P_t = \{(u_t^k, i) \in P \mid \exists (i,j) \in R, serv_{ij} = 0, L_t + d_{ij} \leq V\};\ n_2 = |P_t|;$

pokud $n_2 = 0$ jdi na Krok 7.

Krok 6: {výběr cesty z množiny P_t a přesun vozidla k neobsloužené povinné hraně}

$p_{u_t^k j^*} = \min_{(u_t^k, j) \in P_t} p_{u_t^k j};\ u_{t+1}^k = j^*;\ F_k = F_k + p_{u_t^k u_{t+1}^k};\ t = t+1;$

pokud $F_k \geq F^*$ pak

$k = k+1$; jdi na Krok 2;

jestliže $u_t^k = 1$ nastav $L_t = 0$;

jdi na Krok 3.

Krok 7: {návrat vozidla do depotu}

Jestliže $u_t^k \neq 1$ pak

$u_{t+1}^k = 1; F_k = F_k + p_{u_t^k 1};\ t = t+1;$

$L_t = 0$;

pokud $F_k \geq F^*$ pak

$k = k+1$; jdi na Krok 2;

pokud $n_R = 0$ jdi na Krok 8 jinak jdi na Krok 3.

Krok 8: {test řešení}

Pokud $F_k < F^*$ pak

$F^* = F_k;\ U^* = U_k;$

$k = k+1$; jdi na Krok 2.

Krok 9: Konec.

V Kroku 4 je hrana (u_t^k, j^*) z množiny R_t vybrána náhodně pomocí generátoru pseudo-náhodných čísel, která jsou následně transformována na hodnoty náhodné veličiny s následujícím pravděpodobnostním rozdělením:

1) Rovnoměrné rozdělení – každá hrana je vybrána s pravděpodobností

$$pst_{ij} = \frac{1}{n_1}, \quad (i, j) \in R_t . \tag{5.23}$$

2) Nerovnoměrné rozdělení, pravděpodobnosti jsou odvozeny z velikosti požadavku:

$$pst_{ij} = \frac{d_{ij}}{\sum_{(r,s)\in R_t} d_{rs}}, \quad (i, j) \in R_t . \tag{5.24}$$

3) Nerovnoměrné rozdělení, pravděpodobnosti jsou odvozeny z délky hrany:

$$pst_{ij} = 1 - \frac{c_{ij}}{\sum_{(r,s)\in R_t} c_{rs}}, \quad (i, j) \in R_t . \tag{5.25}$$

Výše uvedený přístup lze aplikovat i v Kroku 6, v němž se vybírá nejkratší cesta k neobsloužené povinné hraně. Namísto nejkratší cesty lze pro výběr cesty použít následující pravděpodobnosti:

$$pst_{ij} = 1 - \frac{p_{ij}}{\sum_{(r,s)\in P_t} p_{rs}}, \quad (i, j) \in P_t . \tag{5.26}$$

Výhodou randomizovaného přístupu je získání velkého počtu tras, u kterých existuje možnost, že získáme lepší řešení než při použití deterministického algoritmu svázaného striktně danými pravidly pro výběr povinných hran, resp. cest k nim vedoucích.

6 Software pro okružní a rozvozní úlohy

Jak vyplývá z celé práce, okružní a rozvozní úlohy, ať již je jejich klasifikace definována jakkoli, patří mezi základní logistické problémy (Brezina, 2003), které v reálné praxi vyžadují speciální přístupy a samozřejmě také speciálně vytvořený programový aparát. Vzhledem k výpočetní náročnosti všech výše uvedených úloh lze jen stěží vystačit i se sebelepším optimalizačním softwarem. V dnešní době se mezi kvalitní a rychlé řešitele řadí především CPLEX od firmy IBM a GUROBI od stejnojmenné firmy. Další jsou pak MOSEK, Conopt, MINOS, IPOPT. Některé firmy nabízejí kromě řešitele i vhodné modelovací prostředí, umožňující mnohdy nejen pohodlný zápis matematických modelů, ale rovněž příjemné prostředí pro prezentaci výsledků, což především v úlohách založených na teorii grafů přináší zajímavé možnosti pro každodenní práci v logistických firmách. Z modelovacích softwarových produktů lze zmínit např. XPRESS Optimization Suite od firmy Fico, MPL od firmy Maximal Software, dále AIMMS, GAMS a AMPL od stejnojmenných firem. Bližší popis jednotlivých produktů neuvádím, neboť informace lze nalézt přímo na webových stránkách příslušných firem a pro tuto práci nejsou signifikantní.

Optimalizační nástroje ve své hrubé či surové podobě nejsou pro většinu firem a jejich plánování tras dostačující, a to především z hlediska speciálních požadavků, které modelování komplikují a prakticky znemožňují, neboť logistické firmy, až na výjimky, většinou nedisponují vlastním analytickým či vývojovým pracovištěm, které by bylo schopné přímo převádět do praxe složité matematické modely a postupy, které jsou předmětem této práce. Mnohem jednodušším a efektivnějším přístupem je pořízení softwaru specializovaného právě na okružní a rozvozní problémy, přičemž ve většině případů není obtížné jednotlivé moduly programů přizpůsobit konkrétním požadavkům firem.

Na trhu existuje celá řada softwarových produktů (Partyka a Hall, 2014), které dokáží uspokojit většinu firem zabývajících se svozem či rozvozem, poskytováním služeb pro přepravu lidí, ať již se jedná o lidi hendikepované (Dial-a-Ride Problem) či zákazníky standardní taxi služby, zajišťováním servisu nejrůznějšího charakteru, a v neposlední řadě také posádky vyjíždějící k naléhavým případům (ambulance, policie, hasiči). V současné době je samozřejmostí propojení plánovacího softwaru s GPS navigací řidiče, který může prakticky ihned reagovat na dynamické změny trasy podle on-line požadavků. Pro úspěšnost každé firmy je důležitý zákazník a obousměrné vazby, které jsou součástí oblasti nazývané Customer Relationship Management (CRM). Pro dokonalé využití specializovaného softwaru je právě tato

část velmi důležitá, neboť schopnost využít veškeré dostupné informace z CRM je nutnou podmínkou pro plánování tras (lze použít i obecnějšího termínu Fleet Management) a v konečném důsledku pro spokojenost zákazníků.

Průzkum, prezentovaný v OR/MS Today (Partyka a Hall, 2014) ukazuje na stále se zvyšující zájem firem o specializovaný software. V přílohách č. 2 – 7 lze nalézt základní údaje o vlastnostech a vybavení vybraných produktů zaměřených na okružní a rozvozní problémy. Jedná se o 17 význačných softwarových produktů firem z Evropy, Severní Ameriky a Asie, které participovaly na dotazníkovém výzkumu.

Podporovaná platforma

Vzhledem k tomu, že ve všech firmách stále převládá operační systém Windows, je podpora této platformy u všech produktů samozřejmostí (viz Příloha č. 2). Není velkým překvapením, že operační systémy UNIX a Linux nejsou, až na výjimky, uvedenými produkty podporovány. Z marketingového hlediska se totiž jedná o marginální cílovou skupinu. Naopak, s rozšiřováním mobilních operačních systémů, především iOS a Android, je pochopitelné, že výrobci soustředí svoji pozornost i k těm zákazníkům, pro které se mobilní zařízení stávají nejpoužívanějším komunikačním prostředkem. Tuto skutečnost lze doložit i faktem, že v předchozím dotazníkovém šetření (Partyka a Hall, 2010) mobilní operační systémy vůbec nefigurovaly, zatímco v současném průzkumu hrají nezanedbatelnou roli. Naproti tomu zůstává stále populární další možnost, jak využívat prostředky na plánování tras, a sice komunikace se softwarem přes webové rozhraní, známá jako Sofware as a Service (SaaS)[76]. Touto službou disponují prakticky všechny vybrané produkty; *Logistics Optimizer* a *Routist.com* se specializují výhradně na tuto platformu.

Použité algoritmy a výkon programu

Všechny produkty, až na výjimky, jsou schopné řešit úlohy s neomezeným počtem výchozích míst, vozidel i obslužných míst. Výpočetní čas závisí samozřejmě na rozsahu úlohy a typu řešeného problému. V Příloze 3 je uveden výpočetní čas pro vybraný rozsah úlohy (50 tras, 1000 míst a dvouhodinová „hard“ časová okna). Údaje jsou samozřejmě pouze orientační a jejich vypovídací schopnost je minimální. Rozhodně by byla zajímavá důkladnější analýza na základě větší variability úloh. Navíc z uvedené tabulky není zřejmé, jak kvalitního

[76] Jednou z nejznámějších služeb tohoto typu v oblasti optimalizace je systém NEOS (Network-Enabled Optimization System).

řešení bylo jednotlivými softwarovými produkty dosaženo. Všechny programy pro řešení používají heuristické a metaheuristické algoritmy a jejich popis je víceméně „černou skříňkou“.

Funkce pro vytváření tras

Zajímavé je porovnání produktů z hlediska základního typu úloh, k jejichž řešení je lze využít (viz Příloha 4). Všechny programy umožňují řešit rozvozní úlohy (Vehicle Routing Problem), šest z nich dokáže navíc vytvářet trasy listonoše (Arc Routing Problem). Pokud jde o časové hledisko vytváření tras, všechny systémy lze použít pro denní plánování a prakticky všechny i pro týdenní plánování. Kromě jednoho produktu jsou všechny schopné vyřizovat on-line požadavky, tedy řešit dynamické úlohy. Zajímavou funkcí je schopnost zahrnout do plánování tras aktuální dopravní informace, stejně tak i schopnost využít údaje o dobách přejezdu mezi místy a dobách zastávek v místech, získaných z reálných tras realizovaných v historii. V Příloze 5 je uveden přehled speciálních funkcí pro vytváření tras, které berou v úvahu geografická omezení, schopnosti a zkušenosti řidiče, a také pravidla, která musí řidič respektovat, jako jsou nutné přestávky na odpočinek, ale také berou v úvahu řidičovy požadavky na přestávku na oběd apod. Opět mají tyto možnosti v sobě integrovány prakticky všechny produkty, stejně jako propojení výsledných tras s mapami a funkcí streetview.

Software jako součást sady s komplexním řešením

Uvedené programy jsou velice často dodávány jako součást sady, která nabízí doplňkové funkce a rozšiřuje možnosti softwaru o další služby (viz Příloha 6). Samozřejmostí je v dnešní době elektronický displej řidiče, jehož součástí je i navigace. Zejména u dynamických úloh hraje důležitou roli možnost lokalizace vozidla; tato funkce, ve spojení s digitálními mapami, může v budoucnu, díky přesnějším informacím a kvalitněji provedené optimalizaci, ušetřit firmám nemalé částky. Systém navíc může obsahovat modul pro evidenci a zpracování objednávek. Především ty firmy, které se zabývají přepravou zásilek, využijí software, který je napojen na RFID skener. Některé produkty jsou součástí systémů pro podporu řízení dodavatelských řetězců. Policie, hasiči a ambulance zase využijí speciální dispečerský systém pro naléhavé případy.

Typ posádky využívající produkt

Klíčovou informací je specifikace, pro koho je software určen či pro jaké účely je daný program vhodný; jedná se tedy o typ posádky, která produkt využívá (viz Příloha 7). Přepravu materiálu, zboží a zásilek lze rozdělit na lokální a dálkovou, přičemž u dálkové lze rozlišit

přepravu menšími nákladními auty a kamiony. Speciálním typem je kurýrní služba. Vzhledem k tomu, že publikovaný výzkum je zaměřen právě na tuto oblast služeb, všechny programy, opět až na výjimky, využívají posádky vozidel zajištujících všechny výše uvedené typy přepravy. Podobně i firmy poskytující např. opravárenské služby, jsou uživateli téměř všech systémů. Jen některé programy jsou využívány speciálním typem přepravy jako je doprava osob autobusy a taxislužbou. Totéž lze říci o řešení naléhavých případů policií, hasiči a ambulancí.

Pokud jde o budoucnost produktů zaměřených na řešení okružních a rozvozních problémů, ta je samozřejmě úzce spojená s teoretickými východisky uvedenými v této práci, ale také s pokrokem v přidružených oblastech jako jsou technika, informační technologie, navigace, apod. Základní aspekty budou zmíněny v závěru práce.

Závěr

Práce je zaměřena na poměrně širokou oblast logistických problémů, a sice okružní a rozvozní úlohy. Pokud by měla práce obsahovat průřez všemi typy problémů a přístupy používanými k jejich řešení, jednalo by se o publikaci o několika dílech, neboť se jedná o problematiku, která je v současné době jednou z nejdynamičtěji se rozvíjejících oblastí logistiky. Toto odvážné tvrzení lze podpořit neuvěřitelně vysokým počtem firem, které se v současné době zabývají svozem a rozvozem, přepravou osob, zboží a zásilek, opravárenskými službami, čištěním komunikací, apod. Jen vyjmenování všech dotčených oblastí by vydalo na několikastránkový seznam. Proto jsem se v práci soustředil pouze na vybrané oblasti mého profesionálního zájmu, a to jak z hlediska vědeckého, tak i z hlediska pedagogického.

Protože většina okružních a rozvozních problémů je založena na úloze obchodního cestujícího (TSP), byla jejímu modelování a metodám pro její řešení věnována celá kapitola. Vzhledem k dalšímu textu jsem se soustředil především na její modifikace související s větším počtem vozidel v jednom či několika výchozích místech a na dynamické rozšíření úlohy, které je spojeno s obsluhou on-line požadavků, přicházejících až po začátku realizace naplánovaných tras. Přestože byly popsány heuristické algoritmy pro řešení úlohy TSP, které jsou poměrně dobře známé a nepovažuji je tudíž za přínos práce, jejich uvedení bylo vhodné především z hlediska modifikace některých algoritmů v dalších úlohách.

Na úlohy s více vozidly jsou zaměřeny i další kapitoly. Jedná se o rozvozní úlohy a úlohu kurýrní služby. V obou případech se v práci zabývám také dynamickým rozšířením těchto problémů. Protože praktické úlohy nabývají mnohdy větších rozměrů, je nutné pro jejich řešení použít heuristické metody. To platí především o úlohách kurýrní služby, kde výpočetní čas hraje klíčovou roli a exaktní přístup je předem vyloučen. Zajímavý je především problém s více vozidly umístěnými v několika výchozích místech s omezenou pracovní dobou. Navržené heuristické postupy mohou být snadno upraveny pro kapacitní úlohy, případně úlohy s časovými okny. Tento záměr je předmětem dalšího výzkumu, v němž budou hrát zásadní roli výpočetní experimenty a porovnání výsledků s jinými heuristickými a metaheuristickými postupy. Výraznou podporu v tomto směru spatřuji v závěrečných pracích studentů navazujícího magisterského studia, a především studia doktorského.

Kapitola o přepravních úlohách, kam byly zařazeny právě i úlohy kurýrní služby, obsahuje také Transhipment Problem, jehož název se často překládá právě jako „přepravní problém“

a bývá často opomíjen pro svoji specifičnost. Z úloh typu „Pickup and Delivery“ (PDP), které jsou popisovány v mnoha publikacích, je v práci uvedena speciální modifikace (zobecněný PDP), v níž se výchozí místa pro jednotlivé okruhy definují až během samotné optimalizace. Výzkum týkající se tohoto problému byl inspirován praktickou úlohou význačné přepravní společnosti.

Cílem této práce je mj. jednoznačně vymezit, že mezi okružní a rozvozní úlohy patří také úloha listonoše (obecně Arc Routing Problem) a její nejrůznější modifikace. Proto byla těmto úlohám věnována samostatná kapitola. Přestože se jedná o úlohy, jejichž řešení lze najít v mnoha publikacích, stále existují praktické modifikace, které vyžadují speciální přístup a které budou předmětem dalšího výzkumu. Jedná se např. o zařazení časových oken, možnost zřízení několika oddělených svozných míst, dynamické úlohy, úlohy s dělenou nakládkou, apod. Je zřejmé, že i v tomto případě se jedná a velmi širokou oblast otevřenou především pro výzkum a vývoj heuristických metod. Některé z nich, včetně upravených verzí, jsou popsány i v této práci.

Samostatná kapitola byla věnována softwarovým produktům vyvinutým speciálně pro okružní a rozvozní úlohy. Přestože záměr zařadit kapitolu do práce tohoto typu může být diskutabilní, považuji za vhodné ukázat, že teoretická východiska zkoumané problematiky mají své vyústění v těch praktických aplikacích, které byly v podstatě inspirací a motivací výzkumu. Přestože prezentovaný průzkum nemá nikterak oslnivé závěry, obsahuje celou řadu zajímavých informací, které jistě budou podnětem pro další výzkum. V době moderních technologií mohou matematické modely a metody využívat informace, které ještě nedávno patřily do oblasti snů. Jedná se například o moduly GPS s přesnou lokalizací vozidla, které ve spojení s digitálními mapami a historickými údaji o realizovaných trasách poskytnou přesným metodám i heuristickým postupům takové vstupní údaje, které umožní firmám najít mnohem efektivnější řešení, než bylo možné získat v minulosti. Stejně tak využívání aktuálních dopravních informací usnadňuje rozhodování dispečerů či přímo řidičů o efektivní změně trasy. Podobně, zařazování nových požadavků do právě realizované trasy, což je předmětem zkoumání dynamických úloh, je dnes mnohem snazší a účinnější díky uvedeným technologiím, což především v případě několika vozidel hraje významnou roli v každodenním boji o zákazníka. V neposlední řadě je u některých firem velice důležitá návaznost tras na další logistické části řetězce, jakými jsou např. zásobovací procesy. Vzhledem k tomu, kolik firem, které řeší své okružní a rozvozní problémy, se v dnešní době pohybuje na trhu, neustále poroste význam

vývoje speciálních softwarových produktů, ale především teoretického výzkumu, zaměřeného na tuto oblast.

V práci jsem se podrobněji zabýval využitím heuristických postupů, nikoli aplikací meta-heuristických algoritmů, ne snad z důvodu, že by se jednalo o okrajovou oblast metodiky, ale proto, že zařazení těchto obecných postupů do této práce ve zkráceném rozsahu nemá očekávaný význam. Jejich zpracování tedy vyžaduje sepsání samostatné práce.

Použitá literatura

[1] Applegate, D. L., Bixby, R. E., Chvátal, V., Cook, W. J.: *The Travelling Salesman Problem – A Computational Study*. Princeton University Press, 2006.

[2] Archetti, C., Speranza, M., Hertz, A.: A Tabu Search Algorithm for the Split Delivery Vehicle Routing Problem. *Transportation Science* **40** (2006), 64–73.

[3] Archetti, C., Mansini, R., Speranza, M. G.: Complexity and Reducibility of the Skip Delivery Problem. *Transportation Science* **39** (2005), 182–187.

[4] Achuthan, N. R., Caccetta, L., Hill, S. P.: An Improved Branch-and-Cut Algorithm for the Capacited Vehicle Routing Problem. *Transportation Science* **37** (2003), 153–169.

[5] Angel, R. D., Caudle, W. L., Noonan, R., Whinston, A.: Computer Assisted School Bus Scheduling. *Management Science* **18** (1972) 279–288.

[6] Assad, A. A., Golden, B. L.: Arc Routing Methods and Applications. 35–139. In Ball, M. O., Magnanti, T. L., Monma, C. L, Nemhauser, G. L. (eds.): *Network Routing, Handbooks in Operations Research and Management Science 8*. Elsevier Science, North-Holland, Amsterdam, 1995.

[7] Ball, M. O., Magnanti, T. L., Monma, C. L, Nemhauser, G. L. (eds.): *Network Routing, Handbooks in Operations Research and Management Science 8*. Elsevier Science, North-Holland, Amsterdam, 1995.

[8] Badeau, P., Gendreau, M., Guertin, F. et al.: A Parallel Tabu Search Heuristic for the Vehicle Routing Problem with Time Windows. *Transportation Science* **5C** (1997), 109–122.

[9] Bard, J. F., Kontoravdis, G., Yu, G.: A Branch-and-Cut Procedure for the Vehicle Routing Problem with Time Windows. *Transportation Science* **36** (2002), 250–269.

[10] Bartásková, P.: *Metoda tvorby tras přepravní úlohy*. Diplomová práce, FIS-VŠE v Praze, 2011.

[11] Bektas, T.: The Multiple Traveling Salesman Problem: an Overview of Formulations and Solution Procedures. *Omega* **34** (2006), 209–219.

[12] Bell, W. J., Dalberto, L. M., Fisher, M. L., Greenfield, A. J., Jaikumar, R., Kedia, P., Macj, R. G ., Prutzman, P. J.: Improving the Distribution of Industrial Gases with On-Line Computerized Routing and Schedulig Optimizer. *Interfaces* **13** (1983), 4–23.

[13] Benavent, E., Corberán, A., Piñana, E., Plana, I., Sanchis, J. M.: New Heuristic Algorithms for the Windy Rural Postman Problem. *Computers and Operations Research* **32** (2005), 3111–3128.

[14] Berbeglia, G., Cordeau, J.-F., Laporte, G.: A Hybrid Tabu Search and Constraint Programming Algorithm for the Dynamic Dial-a-Ride Problem. *INFORMS Journal on Computing* **24** (2012), 343–355.

[15] Berbeglia, G., Pessant, G., Rousseau, L.-M.: Checking the Feasibility Dial-a-Ride Instances Using Constraint Programming. *Transportation Science* **45** (2011), 399–412.

[16] Bertazzi, L., Savelsbergh, M., Speranza, M. G.: Inventory Routing. 49–72. In Golden, B., Raghavan, S., Wasil, E. (eds.): *The Vehicle Routing Problem: Latest Advances and New Challenges*, Springer, 2008.

[17] Bilá, T.: *Svoz směsného odpadu v Poděbradech*. Diplomová práce, FIS-VŠE v Praze, 2013.

[18] Branke, J., Middendorf, M., Noeth, G., Dessouky, M.: Waiting Strategies for Dynamic Vehicle Routing. *Transportation Science* **39** (2005), 298–312.

[19] Bräysy, O., Gendreau, M.: Vehicle Routing Problem with Time Windows, Part I: Route Construction and Local Search Algorithms. *Transportation Science* **39** (2005a), 104–118.

[20] Bräysy, O., Gendreau, M.: Vehicle Routing Problem with Time Windows, Part II: Metaheuristics. *Transportation Science* **39** (2005b), 119–139.

[21] Brezina, I.: *Kvantitatívne metódy v logistike*. EKONÓM, Bratislava, 2003.

[22] Brown, G. G., Ellis, C. J., Lenn, G., Graves, W., Ronen, D.: Real Time, Wide Area Dispatch of Mobil Tank Trucks. *Interfaces* **17** (1987), 107–120.

[23] Cabral, E. A., Gendreau, M., Ghiani, G., Laporte, G.: Solving the Hierarchical Chinese Postman Problem as a Rural Postman Problem. *European Journal of Operational Research* **155** (2004), 44–50.

[24] Calvo, R. W., Cordone, R.: A Heuristic Approach to the Overnight Security Service Problem. *Computers & Operations Research* **30** (2003), 1269–1287.

[25] Cordeau, J.-F.: A Branch-and-Cut Algorithm for the Dial-a-Ride Problem. *Operations Research* **54** (2006), 573–586.

[26] Cordeau, J.-F., Laporte, G.: A Tabu Search Heuristic for the Static Multi-Vehicle Dial-a-Ride Problem. *Transportation Research, Part B* **37** (2003), 579–594.

[27] Cordeau, J.-F., Desaulniers, G., Desrosiers, J., Solomon, M. M., Soumis, F.: VRP with Time Windows. 157–193. In Toth, P., Vigo, D. (eds.): *The Vehicle Routing Problem*. SIAM, 2002.

[28] Cordeau, J.-F., Laporte, G., Ropke, S.: Recent Models and Algorithms for One-to-One Pickup and Delivery Problems. 327–357. In Golden, B. L., Raghavan, S., Wasil, E. A. (eds.): *The Vehicle Routing Problem: Latest advances and New Challenges*. Springer, 2008.

[29] Čičková, Z.: *Aplikácia evolučných algoritmov na riešenie úlohy obchodného cestujúceho*. Dizertačná práca. EU Bratislava, 2005.

[30] Dantzig, G. B., Fulkerson, D. R., Johnson, S. M.: Solution of a Large-Scale Traveling Salesman Problem. *Operations Research* **2** (1954), 393–410.

[31] Desrosiers, J., Dumas, Y., Solomon, M. M., Soumis, F.: Time Constrained Routing and Scheduling. 35–139. In Ball, M. O., Magnanti, T. L., Monma, C. L, Nemhauser, G. L. (eds.): *Network Routing, Handbooks in Operations Research and Management Science 8*. Elsevier Science, North-Holland, Amsterdam, 1995.

[32] Desrochers, M., Desrosiers, J., Solomon, M.: A New Algorithm for the Vehicle Routing Problem with Time Windows. *Operations Research* **40** (1992), 342–354.

[33] Desrochers, M., Laporte, G.: Improvements and Extensions to the Miller-Tucker-Zemlin Subtour Elimination Constraints. *Operations Research Letters* **10** (1991),27–36.

[34] Dial, R. B.: Autonomous Dial-a-Ride Transit Introductory Overview. *Transportation Science* **3C** (1995), 261–275.

[35] Dror, M. (ed.): *Arc Routing: Theory, Solutions and Applications*. Springer, 2000.

[36] Dror, M., Laporte, G., Trudeau, P.: Vehicle Routing with Split Deliveries. *Discrete Applied Mathematics* **50** (1994), 239–254.

[37] Dror, M., Trudeau, P.: Savings by Split Delivery Routing. *Transportation Science* **23** (1989), 141–145.

[38] Dror, M., Stern, H., Trudeau, P.: Postman Tour on a Graph With Precedence Relation on Arcs. *Networks* **17** (1987), 283–294.

[39] Edmonds, J.: The Chinese Postman Problem. *Operations Research* **13** (1965), Supplement 1, B73.

[40] Edmonds, J., Johnson, E. L.: Matching, Euler Tours, and the Chinese Postman. *Mathematical Programming* **5** (1973), 88–124.

[41] Eiselt, H. A., Sandblom, C.-L.: *Integer Programming and Network Models*. Springer, 2000.

[42] Eiselt, H. A., Gendreau, M., Laporte, G.: Arc Routing Problems, Part I: The Chinese Postman Problem. *Operations Research* **43** (1995a), 231–242.

[43] Eiselt, H. A., Gendreau, M., Laporte, G.: Arc Routing Problems, Part II: The Rural Postman Problem. *Operations Research* **43** (1995b), 399–414.

[44] Fábry, J.: Okružní a rozvozní úlohy. 53–84. In Fiala, P. (ed.): *Operační výzkum – nové trendy*. Professional Publishing, Praha, 2010.

[45] Fábry, J.: Capacited Messenger Problem. 84–90. In *Mathematical Methods in Economics 2008* [CD-ROM]. Liberec, 2008.

[46] Fábry, J.: *Dynamické okružní a rozvozní úlohu*. Disertační práce. VŠE-FIS Praha, 2006a.

[47] Fábry, J.: Dynamic Traveling Salesman Problem. 137–146. In *Mathematical Methods in Economics 2006* [CD-ROM]. Plzeň, 2006b.

[48] Fábry, J.: Rozvozní úloha s dělenou dodávkou. In *Progressive Methods and Tools of Management and Economics of Companies* [CD-ROM]. Brno, 2005a.

[49] Fábry, J.: Metoda generování sloupců a její použití v celočíselném programování. 152–163. In Hronová, S. (ed.). *Sborník prací účastníků vědeckého semináře doktorandského studia FIS VŠE v Praze*. Oeconomica, Praha, 2005b.

[50] Fábry, J., Pelikán, J.: Capacitated Postman Problem. 153–158. In *Mathematical Methods in Economics 2013* [CD-ROM]. Jihlava, 2013.

[51] Fábry, J., Kobzareva, M.: Multiple Messenger Problem. 141–147. In *Mathematical Methods in Economics 2012*. Karviná, 2012.

[52] Fábry, J., Pelikán, J.: Method of Column Generation Used in Integer Programming. 242–247. In *Mathematical Methods in Economics 2004*. Brno, 2004.

[53] Fábry, J., Pelikán, J.: Integer Programming and Logic-Based Modeling. 54–60. In *Mathematical Methods in Economics 2003*. ČZU, Praha, 2003.

[54] Favaretto, D., Monetti, E., Pellegrini, P.: Ant Colony System for a VRP with Multiple Time Windows and Multiple Visits. *Journal of Interdisciplinary Mathematics* **10** (2007), 263–284.

[55] Flood, M. M.: The Traveling-Salesman Problem. *Operations Research* **4** (1956), 61–75.

[56] Francis, P. M., Smilowitz, K. R., Tzur, M.: The Period Vehicle Routing Problem and its Extensions. 73–102. In Golden, B. L., Raghavan, S., Wasil, E. A. (eds.): *The Vehicle Routing Problem: Latest advances and New Challenges*. Springer, 2008.

[57] Gavish, B. A Note on the Formulation of the m-Salesman Traveling Salesman Problem. *Management Science* **22** (1976), 704–705.

[58] Gavish, B., Srikanth, K.: An Optimal Solution Method for Large-Scale Multiple Traveling Salesmen Problems. *Operations Research* **34** (1986), 698–717.

[59] Gendreau, M., Guertin, F., Potvin, J.-Y., Taillard, É.: Paralel Tabu Search for Real-Time Vehicle Routing and Dispatching. *Transportation Science* **33** (1999), 381–390.

[60] Gendreau, M., Hertz, A., Laporte, G., Stan, M.: A Generalized Insertion Heuristic for the Traveling Salesman Problem with Time Windows. *Operations Research* **46** (1998a), 330–335.

[61] Gendreau, M., Laporte, G., Semet, F.: The Selective Traveling Salesman Problem. *Networks* **32** (1998b), 263–273.

[62] Gendreau, M., Potvin, J.-Y.: Dynamic Vehicle Routing and Dispatching. 115–126. In Crainic, T., Laporte, G. (eds.): *Fleet Management and Logistics*. Kluwer Academic Publisher, Boston, 1998.

[63] Gendreau, M., Laporte, G., Séguin, R.: Stochastic Vehicle Routing. *European Journal of Operational Research* **88** (1996), 3–12.

[64] Gendreau, M., Hertz, A., Laporte, G.: A Tabu Search Heuristic for the Vehicle Routing Problem. *Management Science* **40** (1994), 1276–1290.

[65] Ghiani, G., Laporte, G., Semet, F.: The Black and White Traveling Salesman Problem. *Operations Research* **54** (2006), 366–378.

[66] Ghiani, G., Improta, G.: An Algorithm for the Hierarchical Chinese Postman Problem. *Operations Research Letters* **26** (2000), 27–32.

[67] Golden, B., Raghavan, S., Wasil, E. (eds.): *The Vehicle Routing Problem: Latest Advances and New Challenges*. Springer, 2008.

[68] Golden, B. L., Wasil, E. A., Kelly, J. P., Chao, I.-M.: Metaheuristics in Vehicle Routing. 33–56. In Crainic, T., Laporte, G. (eds.): *Fleet Management and Logistics*. Kluwer Academic Publisher, Boston, 1998.

[69] Gribkovskaia, I., Laprote, G.: One-to-Many-to-One Single Vehicle Pickup and Delivery Problems. 327–357. In Golden, B. L., Raghavan, S., Wasil, E. A. (eds.): *The Vehicle Routing Problem: Latest advances and New Challenges*. Springer, 2008.

[70] Guan, M.: On the Windy Postman Problem. *Discrete Applied Mathematics* **9** (1984), 41–46.

[71] Guan, M.: Graphic Programming Using Odd or Even Points. *Chinese Mathematics* **1** (1962), 273–277.

[72] Gutin, G. and A. P. Punnen (eds.): *The Traveling Salesman Problem and Its Variations*. Kluwer Academic Publishers, 2002.

[73] Helsgaun, K.: An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristic. *European Journal of Operational Research* **126** (2000), 106–130.

[74] Hertz, A., Laporte, G., Mittaz, M.: A Tabu Search Heuristic for the Capacitated Arc Routing Problem. *Operations Research* **48** (2000), 129–135.

[75] Hill, A., Mabert, V., Montgomory, D.: A Decision Support System for the Courier Vehicle Scheduling Problem. *Omega* **16** (1988), 333–345.

[76] Homberger, J., Gehring, H.: Two Evolutionary Metaheuristics for the Vehicle Routing Problem with Time Windows. *Information Systems and Operational Research* **37** (1999), 297–318.

[77] Chen, Z.-L., Xu, H.: Dynamic Column Generation for Dynamic Vehicle Routing with Time Windows. *Transportation Science* **40** (2006), 74–88.

[78] Christofides, N., Mingozzi, A., Toth, P.: Exact Algorithms for the Vehicle Routing Problem, Based on Spanning Tree and Shortest Path Relaxations. *Mathematical Programming* **20** (1981), 255–82.

[79] Ichoua, S., Gendreau, M., Potvin., J.-Y.: Diversion Issues in Real-Time Vehicle Dispatching. *Transportation Science* **34** (2000), 426–438.

[80] Janáček, J.: *Optimalizace na dopravních sítích*. EDIS, Žilina, 2003.

[81] Jaw, J., Odoni, A. R., Psaraftis, H. N., Wilson, N. H. M.: A Heuristic Algorithm for the Multi-Vehicle Advance-Request Dial-a-Ride Problem with Time Windows. *Transportation Research, Part B* **20** (1986), 243–257.

[82] Jorgensen, R. M., Larsen, J., Bergvinsdottir, K. B.: Solving the Dial-a-Ride Problem Using Genetic Algorithms. *Journal of the Operational Research Society* **58** (2007), 1321–1331.

[83] Kobzareva, M.: *Heuristické algoritmy pro úlohu kurýrní služby*. Diplomová práce. VŠE-FIS, Praha, 2012.

[84] Kolen, A. W. J., Rinnooy Kan, A. H. G., Trienekens, H. W. J. M.: Vehicle Routing with Time Windows. *Operations Research* **35** (1987), 266–273.

[85] Laporte, G., Semet, F.: Classical Heuristics for the Capacitated VRP. 109–128. In Toth, P., Vigo, D. (eds.): *The Vehicle Routing Problem*. SIAM, 2002.

[86] Laporte, G., Nobert, Y., Taillrefer, S.: Solving a Family of Multi-Depot Vehicle Routing and Location-Routing Problems. *Transportation Science* **22** (1988), 161–172.

[87] Laporte, G., Nobert, Y.: A Cutting Planes Algorithm for the m-Salesman Problem. *Operational Research Quarterly* **31** (1980), 1017–1023.

[88] Lenstra, J. K., Rinnooy Kan, A. H. G.: On General Routing Problems. *Networks* **6** (1976), 273–280.

[89] Lin, S.: Computer Solution of the Traveling Salesman Problem. *Bell System Technical Journal* **44** (1965), 2245–2269.

[90] Lin, S., Kernighan, B. W.: An Effective Heuristic Algorithm for the Traveling-Salesman Problem. *Operations Research* **21** (1973), 498–516.

[91] Longo, H., Poggi de Aragao, M. V. S., Uchoa, E.: Solving Capacitated Arc Routing Problems Using a Transformation to the CVRP. *Computers and Operations Research* **33** (2006), 1823–1837.

[92] Lund, K., Madsen, O., Rygaard, J.: *Vehicle Routing Problem with Varying Degrees of Dynamics. Technical report. IMM-REP-1996-1*. Institute of Mathematical Modeling, Technical University of Denmark, Lyngby, Denmark, 1996.

[93] Miller, C. E., Tucker, A. W., Zemlin, R. A.: Integer Programming Formulations and Traveling Salesman Problems. *Journal of the ACM* **7** (1960), 326–329.

[94] Minieka, E.: The Chinese Postman Problem for Mixed Networks. *Management Science* **25** (1979), 643–648.

[95] Okonjo-Adigwe, C.: An Effective Method of Balancing the Workload Amongst Salesmen. *Omega* **16** (1988), 159–163.

[96] Orloff, C. S.: A Fundamental Problem in Vehicle Routing. *Networks* **4** (1974), 35–64.

[97] Papadimitriou, C. H.: On the Complexity of Edge Traversing. *Journal of the ACM* **23** (1976), 544–554.

[98] Papadimitriou, Ch. H., Steiglitz, K.: *Combinatorial Optimization – Algorithms and Complexity*. Dover Publications, 1998.

[99] Partyka, J. H., Hall, R.: Vehicle Routing Software Survey: VR delivers the goods. *OR/MS Today* **41** (2014), 40–47.

[100] Partyka, J. H., Hall, R.: On the Road to Connectivity. *OR/MS Today* **37** (2010), 42–49.

[101] Pelikán, J.: *Diskrétní modely v operačním výzkumu*. Professional Publishing, Praha, 2001.

[102] Pelikán, J., Fábry, J.: Heuristics for Routes Generation in Pickup and Delivery Problem. *Central European Journal of Operations Research* **20** (2012), 463–472.

[103] Potvin, J.-Y., Kervahut, T., Garcia, B.-L., Rousseau, J.-M.: The Vehicle Routing Problem with Time Windows, Part I: Tabu Search. *INFORMS Journal on Computing* **8** (1996), 158–164.

[104] Potvin, J.-Y., Bengio, S.: The Vehicle Routing Problem with Time Windows, Part II: Genetic Search. *INFORMS Journal on Computing* **8** (1996), 165–172.

[105] Powell, W., Jaillet, P., Odoni, A.: Stochastic and Dynamic Networks and Routing. 141–295. In Ball, M. O., Magnanti, T. L., Monma, C. L, Nemhauser, G. L. (eds.): *Network Routing, Handbooks in Operations Research and Management Science 8*. Elsevier Science, Amsterdam, The Netherlands, 1995.

[106] Powell, W., Sheffi, Y., Nickerson, K. S., Butterbaugh, K., Atherton, S.: Maximizing Profits for North American Van Lines Truckload Division: A New Framework for Pricing and Operations. *Interfaces* **18** (1988), 21–41.

[107] Přibylová, L: *Heuristiky pro kapacitní úlohy kurýrní služby*. Diplomová práce. VŠE-FIS, Praha, 2014.

[108] Psaraftis, H. N.: A Multi-comodity, Capacitated Pickup and Delivery Problem: the Single and Two-vehicle Cases. *European Journal of Operational Research* **215** (2011), 572–580.

[109] Psaraftis, H. N.: Dynamic Vehicle Routing: Status and Prospects. *Annals of Operations Research* **61** (1995), 143–64.

[110] Psaraftis, H. N.: Dynamic Vehicle Routing Problems. 223–248. In Golden, B. L., Assad, A. (eds.): *Vehicle Routing: Methods and Studies*. North Holland, Amsterdam, The Netherlands, 1988.

[111] Psaraftis, H. N.: An Exact Algorithm for the Single Vehicle Many-to-Many Dial-a-Ride Problem with Time Windows. *Transportation Research* **17** (1983), 351–357.

[112] Psaraftis, H. N.: A Dynamic Programming Solution to the Single-Vehicle Many-to-Many Immediate Request Dial-a-Ride Problem. *Transportation Science* **14** (1980), 130–154.

[113] Punnen, A. P: The Travelling Salesman Problem: Applications, Formulations and Variations. 1–28. In Gutin, G., Punnen, A. P. (eds): *The Traveling Salesman Problem and Its Variations*. Kluwer Academic Publishers, 2002.

[114] Russel, R. A.: Hybrid Heuristics for the Vehicle Routing Problem with Time Windows. *Transportation Science* **29** (1995), 156–166.

[115] Santos, L., Coutinho-Rodrigues, J., Current, J. R.: An Improved Ant Colony Optimization Based Algorithm for the Capacitated Arc Routing Problem. *Transportation Research, Part B* **44** (2010), 246–266.

[116] Savelsbergh, M. W. P.: The Vehicle Routing Problem with Time Windows: Minimizing Route Duration. *ORSA Journal on Computing* **4** (1992), 146–154.

[117] Savelsbergh, M., Sol, M.: DRIVE: Dynamic Routing of Independent vehicles. *Operations Research* **46** (1998), 474–490.

[118] Solomon, M. M., Desrosiers, J.: Time Windows Constrained Routing and Scheduling Problems. *Transportation Science* **22** (1988), 1–13.

[119] Svestka, J. A., Huckfeldt, V. E.: Computational Experience with an m-Salesman Traveling Salesman Algorithm. *Management Science* **19** (1973), 790–799.

[120] Svobodová, V.: *Optimalizace údržby autobusových zastávek*. Diplomová práce. VŠE-FIS, Praha, 2013.

[121] Taillard, É. D., Badeau, P., Gendreau, M, Guertin, F., Potvin, J.-Y.: A Tabu Search Heuristic for the Vehicle Routing Problem with Soft Time Windows. *Transportation Science* **31** (1997), 170–186.

[122] Toth, P., Vigo, D. (eds.): *The Vehicle Routing Problem*. SIAM, 2002.

[123] Toth, P., Vigo, D.: Exact Solution of the Vehicle Routing Problem. 1–31. In Crainic, T., Laporte, G. (eds.): *Fleet Management and Logistics*. Kluwer Academic Publisher, Boston, 1998.

[124] Toth, P., Vigo, D.: Heuristic Algorithms for the Handicapped Persons Transportation Problem. *Transportation Science* **31** (1997), 60–71.

Internetové zdroje:

[125] http://fausto.eco.unibs.it/dmq/ricerca/quaderni/201.pdf

Archetti, C., Mansini, R., Speranza, M. G.: *The Split Delivery Vehicle Routing Problem with Small Capacity. Technical Report*. Department of Quantitative Methods, University of Brescia, 2001. (31.7.2006).

[126] http://www.idsia.ch/~luca/MAS2003_18.pdf

Gambardella, L. M., Rizzoli, A. E., Oliverio, F., Casagrande, N., Donati, A. V., Montemanni, R., Lucibello, E.: Ant Colony Optimization for Vehicle Routing in Advanced Logistics Systems. *Technical Report*. IDSIA, 2003. (7.7.2014).

[127] ftp://ftp.idsia.ch/pub/techrep/IDSIA-23-02.pdf.gz

Montemanni, R., Gambardella, L. M., Rizzoli, A. E., Donati, A. V.: A New Algorithm for a Dynamic Vehicle Routing Problem Based on Ant Colony System. *Technical Report IDSIA-23-02*. IDSIA, 2002. (7.7.2014).

[128] http://sblocal.aub.edu.lb/PersonalSites/DrOsman/docs/Articles - pdf/VRPTW-GATS.pdf

Thangiah, S. R., Osman, I. H., Sun, T.: Hybrid Genetic Algorithm, Simulated Annealing and Tabu Search Methods for Vehicle Routing Problems with Time Windows. *Working Paper UKC/IMS/OR94/4*, University of Kent, Canterbury, 1994 (7.7.2014).

[129] http://www.fico.com/en/

Internetové stránky společnosti Fico (13.7.2014), jejíž produktem je systém XPRESS-Ive.

Přílohy

Příloha 1 – Matematický model úlohy kurýrní služby s několika vozidly a časovými okny (zápis v optimalizačním systému XPRESS-Ive)

```
model "Capacited Messenger"
 uses "mmxprs", "mmive"

 declarations
  N = 11
  KK=5
  M = 100000000
  Mesto = 1..N
  Trasa = 1..KK

  q: array(Mesto) of integer        ! Pozadavek
  a: array(Mesto) of integer        ! Casove okno
  KM: array(Mesto,Mesto) of integer ! Vzdalenosti
  d: array(Mesto,Mesto) of integer ! Casove prejezdy
  VV: integer                       ! Kapacita vozidla

  x: array(Mesto,Mesto,Trasa) of mpvar   ! Promenne
  v: array(Mesto, Trasa) of mpvar      ! Naklad
  t: array(Mesto,Trasa) of mpvar      ! Cas navstevy

 end-declarations

 initializations from 'vrptw.dat'
  q KM d a VV
 end-initializations

! Ucelova funkce - Celkova dleka vsech tras
 Delka:= sum(i,j in Mesto| i<>j, k in Trasa) KM(i,j)*x(i,j,k)

! Z každeho uzlu se jede prave jednou(krome depotu)
 forall(i in Mesto| i>1) sum(k in Trasa, j in Mesto) x(i,j,k) = 1

! Z depotu vyjizdi na trasu kazde vpzidlo maximalne jednou
 forall(k in Trasa) sum(i in Mesto | i>1) x(1,i,k) <= 1

! Kdyz se vjede do uzlu j na trase k, musi se z nej take na teto trase vyjet
 forall(j in Mesto, k in Trasa) sum(i in Mesto) x(i,j,k) - sum(l in Mesto) x(j,l,k) = 0

! Bilance nakladu
 forall(i,j in Mesto | i<>j and j>1, k in Trasa) v(i,k)+q(j)-2*VV*(1-x(i,j,k)) <= v(j,k)

! Neprekrocit kapacitu vozidla na zadne trase
 forall(i in Mesto| i>1, k in Trasa) v(i,k)<=VV
```

```
! Cas navstivenych zakazniku na trase
forall(i,j in Mesto | i<>j and j>1, k in Trasa) t(i,k)+d(i,j)-M*(1-x(i,j,k))<=t(j,k)

! Vyzvednuti zasilky pred jejim dorucenim
forall(i in Mesto | i>1 and floor(i/2) = (i/2),k in Trasa) t(i,k)<=t(i+1,k)

! Casove okno vyvzvednuti zasilky
forall(i in Mesto | i>1 and floor(i/2) = (i/2), k in Trasa) t(i,k) >= a(i)*SUM(j in Mesto) x(i,j,k)

! Casove okno doruceni zasilky
forall(i in Mesto | i>1 and floor(i/2) = (i/2), k in Trasa) t(i+1,k) <= a(i+1)*SUM(j in Mesto)
x(i+1,j,k)

! Vyzvednuti i doruceni zasilky na stejne trase
forall(i in Mesto | i>1 and floor(i/2) = (i/2), k in Trasa) SUM(j in Mesto) x(i,j,k)= SUM(j in
Mesto) x(i+1,j,k)

! definice vychozich hodnot nakladu a poradi na trase v depotu
forall(k in Trasa) v(1,k) = 0
forall(k in Trasa) t(1,k) = 0

! zakaz smycek u zakazniku
forall(i in Mesto, k in Trasa) x(i,i,k) = 0

! Definice binarnich promennych
forall(i,j in Mesto, k in Trasa) x(i,j,k) is_binary

! Reseni problému
 minimize(Delka)

! Tisk reseni
 writeln("Celkova delka vsech tras=", getobjval)
 forall(k in Trasa, i,j in Mesto)
   if(getsol(x(i,j,k))>0) then
  writeln("x(",i,",",j,",",k,")=",getsol(x(i,j,k)),"     v(",i,",",k,")=",getsol(v(i,k)),"
t(",i,",",k,")=",getsol(t(i,k)))
   end-if
 end-model
```

! Data file for `vrp.mos'

KM: [

0	25	24	59	45	33	37	55	51	16	55
25	0	11	71	70	46	59	61	55	40	80
24	11	0	78	67	38	53	69	63	40	75
59	71	78	0	65	89	81	15	21	49	82
45	70	67	65	0	46	27	71	71	32	16
33	46	38	89	46	0	21	86	83	40	46
37	59	53	81	27	21	0	83	81	35	25
55	61	69	15	71	86	83	0	7	48	87
51	55	63	21	71	83	81	7	0	46	87
16	40	40	49	32	40	35	48	46	0	46
55	80	75	82	16	46	25	87	87	46	0

]

d: [

0	2	2	5	4	3	3	5	4	1	5
2	0	1	6	6	4	5	5	5	3	7
2	1	0	7	6	3	4	6	5	3	6
5	6	7	0	5	7	7	1	2	4	7
4	6	6	5	0	4	2	6	6	3	1
3	4	3	7	4	0	2	7	7	3	4
3	5	4	7	2	2	0	7	7	3	2
5	5	6	1	6	7	7	0	1	4	7
4	5	5	2	6	7	7	1	0	4	7
1	3	3	4	3	3	3	4	4	0	4
5	7	6	7	1	4	2	7	7	4	0

]

a: [0 4 12 3 13 6 10 3 15 4 14]
q: [0 4 -4 6 -6 7 -7 3 -3 3 -3]

VV: 12

Optimální řešení:

1. trasa: 1-8-9-4-5-1

2. trasa: 1-2-3-1

3. trasa: 1-10-6-7-11-1

Celková délka tras = 410

Příloha 2 – Specializovaný software pro okružní a rozvozní úlohy *Podporovaná platforma* /zdroj: Partyka a Hall (2014)/

Produkt	Firma	Rok uvedení na trh	Podporovaná platforma				
			Windows	iOS	Android	Webové aplikace (SaaS)	Ostatní
ArcGIS for TransportationAnalytics	Esri	2012	ano	ano	ano	ano	
Descartes Routing, Mobile & Telematics	Descartes	1995	ano		ano	ano	
DISC	MJC2	1990	ano	ano	ano	ano	UNIX & Linux
eRoute Logistics	WM Logistics	2002	ano		ano	ano	
JOpt.SDK	DNA Evolutions GmbH	2006	ano			ano	
Logistics Optimizer	Runzheimer Inter-national	2005				ano	
logvrp	Netakil	2010	ano	ano	ano	ano	REST + SOAP Web Service API
Optrak vehicle routing software	Optrak Distribution Software Ltd	1992	ano			ano	
ORTEC Routing and Dispatch	ORTECÂ	2003	ano	ano	ano	ano	
Paragon Routing and Scheduling Optimizer	Paragon Software Systems, Inc	1997	ano			ano	
Roadnet Transportation Suite	Roadnet Technologies	1983	ano	ano	ano	ano	
Routist.com	KKT srl	2013				ano	
StreetSync Basic	RouteSolutions	2008	ano			ano	
StreetSync Desktop	RouteSolutions	2005	ano				
StreetSync Pro	RouteSolutions	2011	ano			ano	
TMW Appian DirectRoute	TMW Systems	1996	ano			ano	
TruckStops	MapMechanics	1983	ano				Server nebo PC

Příloha 3 – Specializovaný software pro okružní a rozvozní úlohy *Algoritmy a výkon* /zdroj: Partyka a Hall (2014)/

Produkt	Algoritmy a výkon	
	Výpočetní čas pro úlohu s 50 trasami, 1000 místy a časovými okny	**Typ algoritmů**
ArcGIS for TransportationAnalytics	< 5 min	6+ základních algoritmů
Descartes Routing, Mobile & Telematics	řádově sekundy	holistické a heuristické algoritmy
DISC	řádově sekundy	vlastní
eRoute Logistics	< 4 min	různé heuristiky
JOpt.SDK	< 10 min	metaheuristiky (simulované žíháni a genetické algoritmy)
Logistics Optimizer	< 5 min	genetické algoritmy
logvrp	cca 30 min	metaheuristiky (lokální prohledávání, simulované žíhání), heuristiky
Optrak vehicle routing software	5-20 min	různé heuristiky
ORTEC Routing and Dispatch	řádově minuty	několik algoritmů (v závislosti na typu úlohy)
Paragon Routing and Scheduling Optimizer	cca 2 min	metoda výhodnostních čísel & zlepšující algoritmy
Roadnet Transportation Suite	<30 sec	vlastní heuristiky
Routist.com	< 5 min	vlastní metaheuristiky
StreetSync Basic	řádově minuty	vlastní algoritmy
StreetSync Desktop	řádově minuty	vlastní algoritmy
StreetSync Pro	řádově minuty	vlastní algoritmy
TMW Appian DirectRoute	< 3 min	vlastní algoritmy
TruckStops	< 5 min	vlastní heuristiky

Příloha 4 – Specializovaný software pro okružní a rozvozní úlohy
Funkce pro vytváření tras
/zdroj: Partyka a Hall (2014)/

Produkt	Funkce pro vytváření tras						
	VRP	**ARP**	**Dynamické úlohy**	**Denní trasy**	**Týdenní trasy**	**Plánování tras s využíváním aktuálních dopravních informací**	**Plánování tras s využíváním historických dob přejezdů a zastávek**
ArcGIS for TransportationAnalytics	ano		ano	ano	ano	ano	ano
Descartes Routing, Mobile & Telematics	ano	ano	ano	ano	ano		ano
DISC	ano	ano	ano	ano	ano	ano	ano
eRoute Logistics	ano	ano	ano	ano	ano		ano
JOpt.SDK	ano			ano	ano		
Logistics Optimizer	ano		ano	ano		ano	
logvrp	ano		ano	ano	ano		
Optrak vehicle routing software	ano		ano	ano	ano	ano	ano
ORTEC Routing and Dispatch	ano		ano	ano	ano	ano	ano
Paragon Routing and Scheduling Optimizer	ano	ano	ano	ano	ano		ano
Roadnet Transportation Suite	ano	ano	ano	ano	ano	ano	ano
Routist.com	ano		ano	ano	ano	ano	ano
StreetSync Basic	ano		ano	ano	ano	ano	ano
StreetSync Desktop	ano		ano	ano	ano	ano	ano
StreetSync Pro	ano		ano	ano	ano	ano	ano
TMW Appian DirectRoute	ano		ano	ano	ano		ano
TruckStops	ano	ano	ano	ano	ano	ano	ano

Příloha 5 – Specializovaný software pro okružní a rozvozní úlohy *Speciální funkce pro vytváření tras a jejich mapování* /zdroj: Partyka a Hall (2014)/

Produkt	Speciální funkce pro vytváření tras			Mapování	
	Bere vytváření tras v úvahu schopnosti řidiče	Bere vytváření tras v úvahu geografická omezení	Bere vytváření tras v úvahu řidičova pravidla	Je řešení integrováno s mapami	Je dostupná funkce streetview
ArcGIS for TransportationAnalytics	ano	ano	ano	ano	
Descartes Routing, Mobile & Telematics	ano	ano	ano	ano	ano
DISC	ano	ano	ano	ano	ano
eRoute Logistics		ano	ano	ano	ano
JOpt.SDK	ano	ano	ano		
Logistics Optimizer	ano		ano	ano	
logvrp	ano	ano	ano	ano	ano
Optrak vehicle routing software	ano	ano	ano	ano	ano
ORTEC Routing and Dispatch	ano	ano	ano	ano	ano
Paragon Routing and Scheduling Optimizer	ano	ano	ano	ano	ano
Roadnet Transportation Suite	ano	ano	ano	ano	ano
Routist.com	ano	ano	ano	ano	ano
StreetSync Basic		ano	ano	ano	ano
StreetSync Desktop		ano	ano	ano	ano
StreetSync Pro	ano	ano	ano	ano	ano
TMW Appian DirectRoute	ano	ano	ano	ano	ano
TruckStops	ano	ano	ano	ano	ano

Příloha 6 – Specializovaný software pro okružní a rozvozní úlohy *Software jako součást sady s komplexním řešením* /zdroj: Partyka a Hall (2014)/

Produkt	Produkt je součástí sady, která poskytuje					
	Elektronický displej řidiče	**Lokalizace vozidla**	**Čtečka RFID**	**Software pro dodavatelské řetězce**	**Systém pro zpracování objednávek**	**Dispečerská služba (policie, hasiči, ambulance)**
ArcGIS for TransportationAnalytics	ano	ano				
Descartes Routing, Mobile & Telematics	ano	ano	ano	ano		
DISC	ano	ano	ano	ano	ano	ano
eRoute Logistics	ano	ano				ano
JOpt.SDK						
Logistics Optimizer	ano	ano			ano	
logvrp						
Optrak vehicle routing software		ano				
ORTEC Routing and Dispatch	ano	ano	ano	ano	ano	ano
Paragon Routing and Scheduling Optimizer		ano				
Roadnet Transportation Suite	ano	ano				
Routist.com	ano	ano				
StreetSync Basic	ano	ano	ano			ano
StreetSync Desktop	ano	ano	ano		ano	ano
StreetSync Pro	ano	ano	ano		ano	
TMW Appian DirectRoute	ano	ano			ano	
TruckStops	ano	ano	ano	ano	ano	

Příloha 7 – Specializovaný software pro okružní a rozvozní úlohy *Typ posádky využívající produkt* /zdroj: Partyka and Hall (2014)/

Produkt	Typ posádky využívající produkt							
	Lokální přeprava	**Dálková přeprava (střední a velká vozidla)**	**Dálková přeprava (kamiony)**	**Kurýrní služba**	**BUS**	**Taxi**	**Servisní služby**	**Pohotovost (policie, hasiči a ambulance)**
ArcGIS for TransportationAnalytics	ano	ano		ano	ano		ano	ano
Descartes Routing, Mobile & Telematics	ano	ano					ano	
DISC	ano	ano	ano	ano	ano	ano	ano	
eRoute Logistics	ano	ano	ano	ano	ano		ano	ano
JOpt.SDK	ano	ano	ano	ano			ano	ano
Logistics Optimizer	ano			ano			ano	
logvrp	ano	ano	ano	ano	ano	ano	ano	ano
Optrak vehicle routing software	ano	ano	ano	ano				
ORTEC Routing and Dispatch	ano	ano	ano	ano	ano	ano	ano	ano
Paragon Routing and Scheduling Optimizer	ano	ano	ano	ano			ano	
Roadnet Transportation Suite	ano	ano	ano	ano			ano	
Routist.com	ano						ano	
StreetSync Basic	ano	ano	ano	ano			ano	
StreetSync Desktop	ano	ano	ano	ano			ano	
StreetSync Pro	ano	ano	ano	ano			ano	
TMW Appian DirectRoute	ano	ano		ano			ano	
TruckStops	ano	ano	ano	ano			ano	