УНИВЕРЗИТЕТ У БЕОГРАДУ

МАТЕМАТИЧКИ ФАКУЛТЕТ

Ђорђе С. Стакић

МАТЕМАТИЧКИ МОДЕЛИ И РАЗЛИЧИТИ НАЧИНИ ВИШЕКРИТЕРИЈУМСКЕ ОПТИМИЗАЦИЈЕ У ИНТЕРМОДАЛНОМ ТРАНСПОРТУ

докторска дисертација

Београд , 2022

UNIVERSITY OF BELGRADE

FACULTY OF MATHEMATICS

Đorđe S. Stakić

MATHEMATICAL MODELS AND DIFFERENT WAYS OF APPLYING MULTI-CRITERIA OPTIMIZATION IN INTERMODAL TRANSPORT

doctoral dissertation

Belgrade , 2022

Подаци о ментору и члановима комисије

Ментор:

др Миодраг Живковић, редовни професор, Универзитет у Београду, Математички факултет

Чланови комисије:

др Миодраг Живковић, редовни професор, Универзитет у Београду, Математички факултет

др Владимир Филиповић, редовни професор, Универзитет у Београду, Математички факултет

др Ненад Зрнић, редовни професор, Универзитет у Београду, Машински факултет

др Александар Савић, ванредни професор, Универзитет у Београду, Математички факултет

Датум одбране:

Захвалност

Велику захвалност дугујем свом ментору, професору Миодрагу Живковићу, који је значајно и на много начина допринео квалитету и коначном облику ове докторске дисертације. Од њега сам научио много тога још од прве године студија и ово је најбољи исход подршке коју сам од њега имао. Посебну захвалност дугујем и осталим члановима комисије, професорима Владимиру Филиповићу, Ненаду Зрнићу и Александру Савићу, који су детаљним читањем текста и бројним сугестијама помогли да текст дисертације буде комплетнији и бољи. Велику захвалност дугујем и професору у пензији Душану Тошићу који је значајно помогао око избора теме и око развоја дисертације у ранијим фазама. Професор Ненад Зрнић и Радослав Рајковић су заслужни за моје интересовање за интермодални транспорт и овде им се захваљујем за успешну сарадњу на том пољу. У научном смислу доста ми је помогла и сарадња са Аном Анокић, нарочито на пољу метахеуристичких метода. Захваљујући сарадњи са њом, Зорицом Станимировић и Татјаном Давидовић, научио сам много тога, на чему им се захваљујем. Посебно се захваљујем Татјани Давидовић на коришћењу техничких ресурса Математичког института САНУ, које ми је несебично учинила доступним. Хвала професору Милану Станојевићу на корисним сугестијама везаним за вишекритеријумску оптимизацију.

Овом приликом бих са пуном захвалношћу поменуо и све своје учитеље и професоре на пољу рачунарства и информатике, као и математике, од основне школе па до краја докторских студија. Свако од њих је дао значајан допринос да се моје интересовање на овом пољу развија и одржи до данашњих дана. То су: чланови колектива Математичке гимназије и Математичког факултета, као и учитељ Основне школе у Крупњу Радојко Дробњаковић и наставници Зора Петровић и Миодраг Јовић. У своје математичке учитеље убрајам и свога деду Чедомира - Чеду Стакића који ми је много помогао у првим математичким корацима и кога се са пијететом овде сећам.

Посебну захвалност, на огромној подршци и помоћи, дугујем својој породици: мајци Ради, брату Милошу и снахи Ивани. Отац Саво је, нажалост, преминуо на почетку мојих докторских студија. Захваљујем се и колективу Економског факултета који ме искрено прихватио и подржао.

Наслов дисертације: Математички модели и различити начини вишекритеријумске оптимизације у интермодалном транспорту

Резиме: Интермодални транспорт подразумева саобраћај у којем се користи више врста превоза. Његова заступљеност у пракси је постала све већа. Имајући у виду да се ради, углавном, о великим растојањима, оптимизација је постала значајна у овој области. Подразумевано је да се за превоз користе три стандардне врсте контејнера различитих димензија. У складу са задатим критеријумима, развијени су одговарајући математички модели. На основу модела програмиран је егзактни решавач CPLEX који је за мање вредности улазних параметара успевао да пронађе оптимално решење. За један број модела имплементирана су решења у програмском језику C. Улазни подаци за мање инстанце узимани су из праксе. За тестирање већих инстанци улазни подаци су случајно генерисани из изабраног домена. У првом делу фокус је на тражењу оптималне трасе у транспорту, по задатим критеријумима, који укључују океански и копнени део превоза. Проблем се усложњава повећањем броја бродара, броја успутних лука, као и броја начина превоза на копну. У другом делу рада разматрају се још неки проблеми оптимизације интермодалног транспорта. Ту је већа пажња посвећена појединачним пакетима и разматрају се масе и запремине пакета, потом лимити маса и запремина контејнера. Један од решаваних проблема се односи на распоређивање великог пакета на више контејнера и потом избор оптималног начина расподеле у складу са задатим критеријумима. Други решаван проблем припада збирном контејнерском транспорту и односи се на распоређивање већег броја пакета на контејнере, уз разматрање ограничења масе и запремине. Ту се, такође, тражи оптималан распоред у складу са задатим критеријумима, на пример, укупном минималном ценом. Тако разматран проблем припада хетерогеном и хомогеном векторском паковању контејнера. Направљене су бројне рачунарске имплементације егзактних и приближних метода за посматране моделе. Осмишљене су варијанте методе променљивих околина (енгл. Variable Neighborhood Search - VNS) и методе GRASP (енгл. Greedy Randomized Adaptive Search Procedures) за оптимизацију у збирном контејнерском транспорту. Ове приближне методе су упоређене међусобно, као и са решењима које је добио егзактни решавач CPLEX.

Кључне речи: вишекритеријумска оптимизација, математички модели, метахеуристичке методе, интермодални транспорт, паковање контејнера

Научна област: рачунарство

Ужа научна област: оптимизација

УДК број: 519.87(043.3)

Dissertation title: Mathematical models and different ways of applying multi-criteria optimization in intermodal transport

Abstract: Intermodal transport involves traffic with more than one type of transport. Its presence in practice has become very significant. Bearing in mind that these are mostly long distances, optimization has become important in this area. By default, three standard types of containers of different sizes are used for the transport. In accordance with the given criteria adequate mathematical models have been developed. Based on the model, the exact solver CPLEX was programmed, which succeeds to find the optimal solutions for lesser values of the input parameters. For a number of models, solutions have been implemented in the C programming language. The input data for smaller instances was taken from the practice. To test instances of larger size, the input data is randomly generated from the selected domain. In the first part of this work the main focus is the search for the optimal route in transportation, according to the given criteria, which includes ocean and mainland transport. The problem becomes more complex by increasing the number of shipping companies, the number of side ports, as well as the number of modes of transport on land. In the second part of the paper, additional problems related to the optimization of intermodal transport are considered. More attention is paid to the individual packages by considering the mass and volume of the package, and subsequently the limits of mass and volume of the containers. One of solved problems is related to the deployment of a large pack in several containers, then the selection of optimal allocation in accordance with the set criteria. The second solved problem is from the aggregate container transport and it is related to the deployment of a large number of packages into containers, taking the constraints of mass and volume into consideration. Here we also seek an optimal allocation in accordance with the set criteria, eg. the total minimum price. The problem thus considered to belong to the heterogeneous and homogeneous vector bin packing. The numerous computer implementations of exact and approximate methods for the different models are made. Variant methods of Variable Neighborhood Search (VNS) and GRASP (Greedy Randomized Adaptive Search Procedures) have been designed to optimize the aggregate container transport. These approximation methods were compared with each other as well as with solutions obtained by exact solver CPLEX.

Keywords: multi-criteria optimization, mathematical models, metaheuristic methods, intermodal transport, bin packing

Scientific field: computer science

Scientific subfield: optimization

UDC number: 519.87(043.3)

Садржај

1 Увод

Када се приликом транспорта добара користи комбинација водних, сувоземних или ваздушних путева, такав транспорт назива се интермодалним. У овом раду се проучава и решава два типа проблема оптимизације везаних за интермодални транспорт. У оба случаја полази се од претпоставке да су пакети карактерисани својим масама и запреминама, као и да су различити типови контејнера окарактерисани својим капацитетима у односу на масу и запремину.

У првом случају се полази од једног великог пакета који се може поделити на више контејнера и потом се проналази оптималан начин да се подељени пакет транспортује. У другом случају полази се од више малих пакета који се пакују у контејнере, без дељења самих пакета. Ту се, такође, тражи оптимално решење у складу са задатим циљевима и ова проблематика је позната као збирни контејнерски транспорт. Збирни контејнерски транспорт (енгл. Less than Container Load) се односи на транспорт више мањих пакета обично различитих клијената који могу да се групишу и сместе заједно у контејнер. Овде се тај проблем разматра из угла превозника који има на располгању више контејнера за превоз добијеног скупа пакета.

У вези са збирним контејнерским транспортом нагласак је на проблему оптимизације приликом паковања пакета у контејнере. Разматрани проблем је у литератури познат као хетерогени векторски проблем паковања (енгл. heterogeneous vector bin packing) са два типа ограничења, по маси и запремини. Векторски проблеми паковања се разликују од геометријских који подразумевају просторно распоређивање пакета унутар контејнера. Разлог за избор векторског проблема паковања је двојак, са једне стране такав приступ се користи у пракси код компанија које се баве контејнерским транспортом, а други разлог је што су ова ограничења дефинисана стандардима. Ако би се примењивало искључиво геометријско паковање онда не би било гаранције да је задовољен критеријум везан за масу, тако се уместо појединачних димензија пакета разматра њихова запремина, а разматра се и маса, уз задата ограничења нешто мања од оних која су описана стандардом. Проблем је хетерогени јер могу да се користе различити типови контејнера.

Због сложености хетерогеног проблема за веће инстанце је тешко

одредити егзактно решење, па је развијено неколико варијанти метахеуристичких метода GRASP и VNS које су приказане у раду. Овакав тип проблема је у литератури мање разматран него хомогени случај. Од значаја је рад [1] из 1994. где је модел први пут уведен и потом рад [2] из 2016. године. Приликом решавања проблема паковања примењен је приступ да се сложенији проблеми ослањају на решења једноставнијих. То конкретно значи да се прво пронађу трасе за превоз контејнера оптималне по цени, које се потом користе код решавања сложенијих оптимизационих проблема. На пример, код проблема паковања минималне цене превоза контејнера су улазни подаци за даље рачунање.

Осим хетерогеног разматра се и варијанта проблема, када се све пакује у један тип контејнера. Такав вид паковања се зове хомогено векторско паковање и у литератури је више разматран. За хомогено векторско паковање постоји библиотека од 400 инстанци описана у раду [3]. У више радова су предложени методи тестирани на овој библиотеци. Највећи помак је постигнут у раду [4] у којем је оптимално решено 330 инстанци, док за 70 инстанци алгоритам није успео да генерише решење. Постоји и новија библиотека од 400 других великих хомогених инстанци описана у раду [64] међу којима за 61 инстанцу уопште није било познато оптимално решење него само границе где се оно налази. Методе GRASP и VNS развијене за нехомогени случај прилагођене су за решавање хомогеног случаја. Резултати су упоређени како са тачним резултатима добијеним помоћу решавача CPLEX, тако и са резултатима познатим из литературе. Ови резултати за методу GRASP објављени су у раду [5], а за методу VNS објављени су у раду [63].

Дисертације има укупно шест глава. После прве, уводне главе, следи друга глава где је детаљније представљена проблематика тако што је дат опис проблема и циљева оптимизације, основних појмова математичких модела као и приказ стања у области интермодалног транспорта кроз приказ одговарајуће литературе. После тога следи трећа глава у којој се разматра проблем расподеле великог пакета на више контејнера. Ту је направљено неколико варијанти модела, у зависности да ли се позивају на раније одређене оптималне трасе по задатом критеријуму, или их тада истовремено рачунају. У зависности од начина рачунања загађења направљена су два различита модела. У истој трећој глави се рачунају ефикасни и Парето скуп за оптимизацију контејнерског транспорта по цени и загађењу.

У четвртој глави се разматра проблематика збирног контејнерског транспорта у којој је за дати скуп пакета потребно пронаћи оптимално распоређивање у контејнере за задату функцију циља. То се односи на хетерогено векторско паковање контејнера. У петој глави је обрађен специјални случај паковања који се односи на хомогено векторско паковање контејнера. У 4. и 5. глави се приказују примењене метахеуристичке методе GRASP и VNS и решавање помоћу решавача CPLEX. Потом следе закључна 6. глава, додатак и списак коришћене литературе.

Главни доприноси аутора се односе на развијене варијанте метахеуристичких метода GRASP и VNS које су примењене на хетерогено и хомогено векторско паковања у збирном контејнерском транспорту. Најбоље резултате за хомогени случај постигла је варијанта методе VNS која је оптимално решила 60 инстанци од 70 оних које нису решене у раду [4], решавајући укупно 365 од 400 инстанци описаних у раду [3]. Иста метода нашла је 10 оптималних решења, до тада непознатних, као и још четири боље горње границе решења на скупу нерешених инстанци описаних у раду [64]. Део доприноса чини и креирана библиотека од 6 малих и 50 великих инстанци за хетерогено векторско паковање која је учињена доступном истраживачима.

2 Опис проблема и стање у области

Интермодални контејнерски транспорт има значајан удео у транспорту на светском нивоу. У наставку се описани одговарајући математички модели који добро одсликавају природу интермодалног транспорта и на којима се добро могу сагледати проблеми који се овде јављају. Осим тога, циљ је да се испита како различитим методама оптимизације такав вид транспорта учинити што ефикаснијим.

2.1 Опис разматраног проблема

Разматра се следећи транспортни сценарио:

Претпоставимо да постоји полазна лука и одредиште унутар копна. Без умањења општости, може се претпоставити да је у питању прекоокеанска лука. Претпоставимо да се део транспорта одвија преко океана и да има више могућих лука за претовар ради наставка пута копном. Како од полазне луке до циља не може да се стигне без претоварања и истом врстом транспорта, у питању је интермодални транспорт. Од полазне до претоварне луке могу да постоје успутне луке које се ради поједностављења неће разматрати. Од последње луке до циља се може стићи копном коришћењем више врста транспорта: камион, железница, река. Преко океана, од полазне луке до претоварних лука путује се искључиво прекоокеанским бродовима. На путу од одредишта до циља користи се увек једна од неколико могућих претоварних лука.

Формулација проблема се усложњава јер обично постоји више бродара који обављају транспорт преко океана. Сваки од њих има своје услове транспорта, пре свега цену, али и предвиђено временско трајање путовања, као и растојање које се прелази. Бродари имају различите цене, у зависности од типа контејнера и претоварне луке. Осим тога, сваки бродар до сваке претоварне луке може да понуди више различитих траса које ћемо овде назвати сервиси. Сервиси истог бродара до исте претоварне луке обично имају исту цену, али се могу разликовати по другим карактеристикама, пре свега по успутним лукама, а тиме и по дужини трајања пута и растојању што може да утиче на загађење.

У делу транспорта од претоварних лука до циља на копну може се сматрати да цена, време и дужина пута, зависе само од луке, врсте

транспорта, типа контејнера и одредишта. Ради поједностављења модела може се сматрати да на копненим деоницама не постоји више понуђача услуга за дату врсту транспорта, за разлику од више бродара на поморском делу транспорта.

Заједнички приступ у решавању проблема у наредним поглављима је да се за сваки тип контејнера од полазне луке до циља претходно одреди оптимална траса у односу на изабрану циљну функцију.

2.1.1 Карактеристике контејнера

Према ISO 668:1995 стандарду, описаном у [29], димензије три основне врсте контејнера и њихови капацитети у погледу дозвољене носивости масе и запремине налазе се у табели 1.

	20' конт. (1)	40' конт. (2)	40' високи конт. (3)
спољна дужина	$6,058m$	$12,192m$	$12,192m$
спољна ширина	$2,438m$	$2,438m$	$2,438m$
спољна висина	$2,591m$	$2,591m$	$2,896m$
унутр. дужина	$5,867m$	$12,032m$	$12,000m$
унутр. ширина	$2,352m$	$2,352m$	$2,311m$
унутр. висина	$2,385m$	$2,385m$	$2,650m$
унутр. запремина	$33,1m^3$	$67,5m^3$	$75,3m^3$
максимална маса	$30400kg$	$30400kg$	$30848kg$
маса празног	$2200kg$	$3800kg$	$3900kg$
нето носивост	$28200kg$	$26600kg$	$26580kg$

Табела 1: Димензије, лимити маса и запремина контејнера

Контејнери описани у првим колонама су 20-стопни јер им је спољна дужина приближно 20 стопа. Контејнери у другој и трећој колони су 40-стопни јер им је спољна дужина 40 стопа. Разлике између ширина све три ове врсте контејнера не постоје. Ове три врсте контејнера се често означавају редом са $20DV$, $40DV$ и $40HQ$. *DV* је скраћено од енглеског назива *dry van*, а *HQ* од *high cube*. Разлике у висини 20-стопних и обичних 40-стопних контејнера не постоје. Типовима контејнера су у овом раду додељени редни бројеви, наведени су у заградама у табели 1.

У имплементацији је узето да су лимити нешто мањи од стандарда, јер се тако користи и у пракси, пошто није пожељно да се иде до максимално могућих маса и запремина које дефинише стандард. Ове вредности су наведене у табели 2.

	20' конт.	40' конт.	40' високи конт.
запремина	$30m^3$	$60m^3$	$70m^3$
маса	$25,8t$	$24,5t$	$24,5t$

Табела 2: Коришћени лимити маса и запремина

Скраћеница *TEU* (енгл. Twenty-foot equivalent unit) се користи за стандардни 20-стопни контејнер.

2.2 Математички модели - основни појмови

Када се разматра реални проблем у некој области, потребно је установити његове основне особине и представити их на унификован и прихватљив начин. Математичко моделирање даје оквир за представљање проблема који се решава. Најчешће се тај оквир користи да се издвоји одговарајућа функција, такозвана функција циља, коју је потребно оптимизовати. Под оптимизацијом овде се подразумева одређивање њене максималне или минималне вредности.

Модел углавном зависи од више променљивих које у њему фигуришу. Неке од променљивих се добијају као његови улазни подаци, који су познати на почетку решавања проблема. Обично их зовемо улазни параметри. Од њих зависи функција циља, као и све остало што се у моделу наводи.

Осим улазних података и функције циља, значајан део математичког модела су ограничења. Ограничењима се детаљније описује разматрани проблем и његове специфичности. Неким ограничењима се дефинише једнакост која је потребно да важи, док се у неким ограничењима наводе неједнакости. Код навођења неједнакости обично се не користе строге неједнакости, него $\leq$ и $\geq$, да би простор у којем се траже решења био затворен скуп и као такав достизао одређене екстремне вредности.

У математичком моделу, осим познатих улазних података, појављују се и променљиве одлучивања (енгл. decision variable). Ове променљиве могу да буду целобројне или реалне, а некада су и бинарне, тј. узимају вредности из скупа $\{0, 1\}$. За различите вредности променљивих одлучивања добијају се различите вредности функције циља. Потребно је одредити за које вредности променљивих одлучивања, из дозвољеног домена, функција циља достиже своју екстремну вредност, а при том су

задовољена и сва наведена ограничења.

Једном направљен математички модел може имати велику примену јер има универзални карактер за проблем који се разматра. У том случају, када се замене улазни подаци, добија се потпуно нова екстремна вредност функције циља за нове вредности променљивих одлучивања.

Математички модели се могу класификовати на линеарне и нелинеарне, међу којима се посебно издвајају квадратни. Код линеарних модела сви изрази, у ограничењима и функцији циља, су линеарни у односу на непознате променљиве. Код квадратних модела су, осим линеарних, дозвољени и изрази са квадратним факторима непознатих променљивих. Сматра се да је модел нелинеаран уколико је бар један његов израз нелинеаран у зависности од непознатих променљивих.

Развијање математичког модела представља прву фазу решавања задатог проблема оптимизације. У тој фази се реална слика апроксимира направљеним моделом. Потом се иде на следећу фазу у којој се, у зависности од вредности улазних података, решава задатак за који је тај модел креиран. У решавању могући су многи приступи који се грубо могу поделити на два основна. Један приступ је егзактно решавање. То подразумева да се добије решење које је гарантовано оптимално. Зависно од облика модела и димензија његових улазних података, ово је некад мање, некад више, тежак задатак. Постоје егзактни решавачи оспособљени да помоћу својих уграђених функција решавају проблеме одређених класа и до одређених димензија. Ово обично захтева да се запис самог модела са математичких формула прилагоди и запише на језику који конкретан решавач може да разуме. Неки од најчешће коришћених решавача су Lingo и CPLEX. У овом раду је мањи број модела тестиран у систему Lingo 17.0. Иако је у питању комерцијалан софтвер, могуће је добити бесплатну академску лиценцу за образовне и научно-истраживачке сврхе, па је та погодност искоришћена и за овај рад. Већина тестирања урађена је помоћу решавача CPLEX 12.6.2. Осим готових решавача, за поједине једноставније моделе могуће је егзактно решење добити и директним програмирањем решења у неком окружењу и језику. За те сврхе овде је коришћен програмски језик C.

Други правац коришћења датог математичког модела, посебно када није могуће једноставно пронаћи или гарантовати егзактно решење, је налажење довољно доброг допустивог решења. Код приближног реша-

вања су развијене бројне метахеуристичке методе. Неке од метода су инспирисане појавама из природе, као на пример оптимизација колонијом мрава (енгл. Ant colony optimization), или колонијом пчела (енгл. Bee colony optimization) и друге. Једна од најпознатијих класа метахеуристичких метода су генетски алгоритми инспирисани еволутивним процесима у биологији. Затим, ту је метода промењливих околина (VNS), табу претраживање, методе засноване на похлепном алгоритму и случајном избору као што је GRASP и бројне друге. Свакодневно се усавршавају постојеће и проналазе нове метахеурустичке методе којима се решавају многи проблеми оптимизације. У овом раду су развијане метахеуристичке методе VNS и GRASP за решавање одређених проблема интермодалног транспорта. Оне су програмиране у програмском језику C.

2.3 Циљеви оптимизације

У интермодалном транспорту оптимизација се може вршити по разним критеријумима. Ако се посматра једна циљна функција, реч је о једнокритеријумској оптимизацији. Посебан приступ захтева оптимизација по два, три или више циљева и тада је реч о вишекритеријумској оптимизацији.

Приликом оптимизације интермодалног транспорта прво се одаберу циљне функције. У овом раду изабрани су цена, време и утицај на животну средину - загађење. Тај избор је очекиван јер се и у пракси углавном они разматрају.

У процесу транспорта од полазне луке до циља, присутни су бројни трошкови. У укупну цену трошкова пута улазе: трошкови превоза од полазне до претоварне луке, лучки трошкови и трошкови превоза од претоварне луке до циља. У пракси се ови трошкови могу јављати у различитим валутама, то је овде поједностављено и конвертовано у једну валуту.

Претпоставља се да је време изражено у данима. Разлог за то је, пре свега, зато што су у питању велика растојања. Разлози важности времена некад су везани за саму робу, њен рок трајања или осетљивост, или су комерцијалне природе, јер краће време рада са једном туром робе повећава број могућих тура у неком периоду па се и на тај начин повећава (или смањује) профит.

Трећи параметар, који је узет у разматрање, је утицај на животну средину, односно загађење. Повећан саобраћај узрокује повећану емисију угљен-диоксида (CO_2). Овај гас се сматра загађивачем који утиче на озонски омотач и на климатске промене. Овде се разматрају укупно четири врсте транспорта: прекоокеански брод, железница, камион и транспорт реком. За сваки од њих постоје одговарајући коефицијенти загађења по пређеном километру, који су познати из литературе. У радовима се обично користе коефицијенти загађења изражени у јединици $kg/TEUkm$. Њихове вредности коришћене у раду [6] наведене су у табели 3.

врста транспорта	коефицијент загађења
брод	0,084
пруга	0,205
баржа	0,084
камион	0,472

Табела 3: Вредности коефицијената загађења

Наведени коефицијенти се користе и овде, осим у ретким случајевима када је то наглашено.

2.4 Преглед релевантне литературе

Проблематика обухваћена у овом раду је савремена и актуелна за истраживаче различитог профила, зато може да јој се приступа из више углова и из различитих научних дисциплина.

Интермодални транспорт је описан у радовима [7, 8]. Истраживачи, који се баве оптимизацијом у интермодалном транспорту, у великом броју радова фокусирају се на конкретне проблеме и задатке, најчешће везано за одређену географску област на којој се одвија тај транспорт. Тако је, на пример, у раду [9] разматрана оптимизација у интермодалном транспорту везана за Европу. У погледу оптимизације у неким радовима се посебна пажња посвећује одређеној врсти транспорта, на пример камионима, возовима, бродовима и слично. Тако је у раду [10] разматран проблем транспорта који укључује само железницу и камион. У погледу оптимизација у различитим радовима посвећује се пажња различитим параметрима. Тако се у радовима [11, 12] анализира оптимизација по цени и загађењу. Од посебног значаја је прегледни рад [13] из 2013. године где су класификовани и разврстани дотадашњи радови

из ове области. Ту се такође истиче важност математичких модела, као и очекивање да ће у будућим радовима већи значај имати двокритеријумска и вишекритеријумска оптимизација. Од значаја је и прегледни рад [14] који даје опсежан преглед литературе у области планирања у овој врсти транспорта.

У погледу математичко-рачунарског оптимизационог апарата у радовима се углавном креирају математички модели који приближно, колико год је то могуће, описују разматрани проблем. У већини радова који у свом фокусу имају интермодални транспорт, разматрају се проблеми који су везани за конкретан саобраћај у реалној ситуацији на терену. Међутим, од интереса су и истраживања везана за методологију која се може универзално применити невезано за конкретан реалан случај. О примени метахеуристичких метода у логистици и транспорту може се више видети у раду [15]. Ту је дат преглед литературе где се види да су најчешће коришћене методе табу претраживање (52 рада), генетски алгоритам (46), симулирано каљење (енгл. Simulated annealing) (23), колонија мрава (8), GRASP (5) и остале методе (9 радова). У поменутом раду [9] од оптимизационих метода коришћена је табу претрага. О метахеуристчким методама примењеним на вишекритеријумску оптимизацију у транспорту може се видети у радовима [16, 17]. Хеуристички и егзактни алгоритми за решавање проблема рутирања возила представљени су детаљно у раду [18].

Током истраживања која су претходила овом раду објављено је више радова који се баве налажењем оптималних траса по једном или више параметара. У њима су углавном разматрани реални улазни подаци контејнерског транспорта између Далеког истока и Србије, мада се методе рада могу применити на било који сличан транспортни сценарио. У раду [19], који је представљен на конференцији у Цељу, а потом и штампан у часопису, разматрана је проблематика налажења оптималне трасе само по цени. У раду [20] је такође разматрана само цена, при чему су коришћена три приступа, а имплементација је била у окружењима Lingo и MATLAB. У раду [21] разматран је проблема налажења оптималне трасе само по времену, а у раду [22] само по загађењу. У радовима [23] и [24] имплементирано је налажење Парето скупова за сваки од три пара параметара цена-време, цена-загађење и време-загађење, као и Парето скупа у три димензије за сва три параметра.

У раду [25] разматране су трасе по цени и времену. Направљен је нешто детаљнији приступ који садржи лучке трошкове, као и курсну листу, тј. однос валута евро-долар (енгл. exchange rate). Насупрот томе, у овом раду су ти детаљи изостављени и сматрају се делом припреме улазних података код којих се све цене конвертују у једну валуту, углавном евро, а лучки трошкови се сматрају укљученим у задату цену транспорта. У раду [25] посматрано је неколико сценарија виђених из угла практичних примена. Као један од приступа примењено је скалирање по цени и времену без коришћења тежинских коефицијената. У раду [26] се такође разматрају цена и време, с тим што је коришћена верзија скалирања са тежинским коефицијентима. У раду [27] анализирана су сва три параметра: цена, време и загађење. На неки начин тропараметарски приступ представља уопштење рада [25] у којем се разматрају само цена и време. Направљено решење имплементирано је у MATLAB-у и у том раду је презентована варијанта која користи скалирање без тежинских коефицијената, односно са једнаким утицајем сва три параметра. У раду [28] примењен је сличан приступ, с тим што је дат осврт и на друге радове из литературе у којима се разматрају оптимизације у транспорту. У свим овим радовима је презентовано и тумачење конкретних резултата, тј. какав је закључак из тога изведен за саобраћај на посматраној релацији.

3 Расподела великог пакета на контејнере

Задатак који је у овом делу разматран односи се на превоз великог пакета, задате масе m и запремине V, који се може поделити на више контејнера тако да се поштују лимити маса и запремина сваког контејнера као и једнак укупан збир. Овде се подразумева да је сваки контејнер, осим евентуално последњег додатог, попуњен до лимита своје масе или лимита своје запремине. На тај начин се, неком врстом компресије, смањује број потребних контејнера. Због различитих димензија, добија се различит број потребних контејнера за превоз задатог пакета. Такође због постојања различитих траса са различитим вредностима циљних функција проблем се додатно усложњава.

У наставку је описано више математичих модела који моделирају овај проблем водећи рачуна о специфичностима постављеног проблема интермодалног транспорта. Као циљне функције су узете у обзир цена и загађење, при чему су загађење разматрана два начина обрачунавања. Посебну целину чини вишекритеријумска оптимизација код које су одређивани ефикасни и Парето скупови имајући у виду обе циљне функције.

Након тога је разматран поједностављен модел који рачуна оптималну цену превоза скупа контејнера на који је распоређен задати велики пакет узимајући у обзир да је претходно за сваки тип контејнера одређена оптимална траса и њој придружена оптимална цена. Приликом распоређивања пакета на контејнере коришћена су стандардна три типа контејнера интермодалног транспорта.

Једно виђење овог приступа је да се проблем анализира из угла корисника услуге транспорта. Претпоставља се да корисник задаје величину свог пакета наводећи његову масу и запремину, а потребно је да се прво нађе расподела како најпогодније да се његов пакет распореди на контејнере, а потом да од свих могућих сценарија изабере онај који је оптималан по задатим параметрима. У овом случају потребно је да сваки од три типа контејнера задовољава сопствена ограничења дозвољене масе и запремине контејнера. Ради поједностављења, може се сматрати да је пакет равномерне густине и да се може раздвајати на контејнере ради транспорта.

3.1 Математички модел

У наставку се наводи један сложенији модел за раније описани транспортни сценарио у интермодалном транспорту. Разматра се истовремени превоз више контејнера који превозе велики пакет, с тим што у овом моделу оптималне трасе нису унапред познате него их он истовремено рачуна. У том моделу се разматрају цена и загађење.

Ради једноставнијег записа модела користиће се ознака $[n] = \{1, \ldots, n\}$ и следеће ознаке у индексима:

t - редни број типа контејнера,

i - редни број бродара,

j - редни број претоварне луке,

k - ознака врсте транспорта на копну,

s - редни број сервиса

Улазни подаци су:

ni - број бродара

nj - број претоварних лука

nk - број врста превоза по копну

ns - могући број сервиса једног бродара до исте луке

nt - број типова контејнера

nl_t - број контејнера типа t, где је $t \in [nt]$

cf_{ijt} - цене на морском делу пута, где су $i \in [ni]$, $j \in [nj]$, $t \in [nt]$

cs_{jkt} - цене превоза на копненом делу пута, где су $j \in [nj]$, $k \in [nk]$, $t \in [nt]$

df_{ijs} - дужина морског дела пута, где су $i \in [ni]$, $j \in [nj]$, $s \in [ns]$

ds_{jk} - дужина копненог дела пута, где су $j \in [nj]$, $k \in [nk]$

ef_t - коефицијент загађења преко океана, где је $t \in [nt]$

es_{kt} - коефицијенти загађења путовања копном, где су $k \in [nk]$, $t \in [nt]$

V - запремина пакета

m - маса пакета

LV_t - горња граница запремине контејнера типа t, где је $t \in [nt]$

Lm_t - горња граница масе контејнера типа t, где је $t \in [nt]$

Промењљиве избором чијих вредности треба оптимизовати циљну функцију, у литератури често назване управљачке или промењљиве одлучивања (енгл. decision variables) су:

f_{ijst} - број контејнера на морском делу пута, где су $i \in [ni]$, $j \in [nj]$, $s \in [ns]$, $t \in [nt]$

s_{jkt} - број контејнера на копненом делу пута, где су $j \in [nj]$, $k \in [nk]$, $t \in [nt]$

p_{lt} - бинарна промењљива која говори да ли се превози $l-$ти контејнер типа t, где су $t \in [nt]$, $l \in [nl_t]$

m_{lt} - маса у $l-$том контејнеру типа t, где су $t \in [nt]$, $l \in [nl_t]$

Одговарајућа ограничења би била:

$$\sum_{i=1}^{ni}\sum_{j=1}^{nj}\sum_{s=1}^{ns} f_{ijst} = \sum_{l=1}^{nl_t} p_{lt}, \quad t \in [nt] \tag{3.1}$$

$$\sum_{i=1}^{ni}\sum_{s=1}^{ns} f_{ijst} = \sum_{k=1}^{nk} s_{jkt}, \quad j \in [nj], t \in [nt] \tag{3.2}$$

$$\sum_{t=1}^{nt}\sum_{l=1}^{nl_t} m_{lt} = m \tag{3.3}$$

$$m_{lt} \leq Lm_t \cdot p_{lt}, \quad t \in [nt], l \in [nl_t] \tag{3.4}$$

$$\frac{V}{m} m_{lt} \leq LV_t, \quad t \in [nt], l \in [nl_t] \tag{3.5}$$

$$p_{lt} \in \{0, 1\}, \quad t \in [nt], l \in [nl_t] \tag{3.6}$$

$$f_{ijst} \in \mathbb{N}_0, \quad i \in [ni], j \in [nj], s \in [ns], t \in [nt] \tag{3.7}$$

$$s_{jkt} \in \mathbb{N}_0, \quad j \in [nj], k \in [nk], t \in [nt] \tag{3.8}$$

$$m_{lt} \geq 0, \quad t \in [nt], l \in [nl_t] \tag{3.9}$$

Са овако уведеним ознакама може се дефинисати цена као циљна функција чију вредност треба минимизовати:

$$C(f,s) = \sum_{i=1}^{ni}\sum_{j=1}^{nj}\sum_{s=1}^{ns}\sum_{t=1}^{nt} cf_{ijt}f_{ijst} + \sum_{j=1}^{nj}\sum_{k=1}^{nk}\sum_{t=1}^{nt} cs_{jkt}s_{jkt} \tag{3.10}$$

Загађење, као алтернативна циљна функција, може се дефинисати на следећи начин:

$$E(f,s) = \sum_{i=1}^{ni}\sum_{j=1}^{nj}\sum_{s=1}^{ns}\sum_{t=1}^{nt} df_{ijs}ef_t f_{ijst} + \sum_{j=1}^{nj}\sum_{k=1}^{nk}\sum_{t=1}^{nt} ds_{jk}es_{kt}s_{jkt} \tag{3.11}$$

Ограничење (3.1) односи се на то да је за сваки тип контејнера број контејнера, који се крећу морским делом пута, једнак укупном броју контејнера тог типа који су узети за разматрање у моделу. Ограничење (3.2) односи се на то да је за сваку луку и сваки тип контејнера број контејнера, који су у њу дошли, једнак броју контејнера који су одатле отишли ка циљу. Ограничење (3.3) односи се на то да укупна маса по контејнерима мора да буде једнака маси полазног пакета. Како је узето у обзир да је густина великог пакета ($\rho = m/V$) константна, онда се, на основу издвојене масе у неком контејнеру, зна и колика је запремина тог дела пакета. Ограничење (3.4) односи се на то да маса у сваком контејнеру не прелази дозвољену масу за тај тип контејнера. Слично, (3.5) односи се на то да запремина дела пакета у сваком контејнеру не прелази лимит запремине за тај тип контејнера. Ограничење (3.4) обезбеђује да, уколико маса у неком контејнеру није 0, онда он мора бити ангажован. Такође, ако неки контејнер није ангажован, онда у њему маса мора да буде 0. Овде није наведено ограничење које спречава, када је маса у контејнеру 0, да p_{lt} не буде 0, али су због рачунања минимума функција C и E фаворити случајеви са мањим бројем заузетих контејнера, па то ограничење није неопходно.

Уместо ограничења (3.1) може се узети ограничење:

$$\sum_{j=1}^{nj}\sum_{k=1}^{nk} s_{jkt} = \sum_{l=1}^{nl_t} p_{lt}, \quad t \in [nt] \tag{3.12}$$

које се односи на то да је за сваки тип контејнера број контејнера, који се крећу другим делом пута (копном), једнак укупном броју контејнера тог

типа који су ангажовани у моделу. Није потребно узимати истовремено оба ограничења (3.1) и (3.12), довољно је једно од њих, јер је ограничењем (3.2) дефинисано слагање збирова у успутним лукама, па ће онда и укупан збир да буде исти.

Приметимо да ограничење облика:

$$\sum_{t=1}^{nt} \sum_{l=1}^{nl_t} \frac{V}{m} m_{lt} = V \tag{3.13}$$

које би се односило на то да укупна запремина по контејнерима мора да буде једнака запремини полазног пакета, није потребно јер је еквивалентно ограничењу (3.3).

Разматрани модел представља комбинацију, са једне стране, проблема рутирања возила (енгл. Vehicle Routing Problem), и са друге стране, проблема паковања који је овде близак проблему ранца. Када су у питању вишеструка ограничења код проблема ранца, то постаје NP-тежак проблем. Слично је и проблем рутирања возила NP-тежак, зато то важи и за њихову комбинацију која својим бројним ограничењима отежава цео проблем.

О проблему рутирања возила са ограничењима при утовару може се више видети у радовима [38] и [39]. Прегледни рад [40] у коме су тема ограничења при утовару контејнера, даје класификацију проблема у овој области. О вишедимензионом проблему ранца може се више пронаћи у прегледном раду [41].

3.1.1 Модификован математички модел

У претходно постављеном моделу нешто је теже дефинисати функцију циља за загађење која би узела у обзир рачунања загађења које има у виду величину пакета. Наиме, у литератури се, осим усредњених коефицијената загађења за поједине типове контејнера по километру за различите типове транспорта који су до сада овде коришћени, користе и коефицијенти који су изражени у јединици gCO_2/tkm; тада на укупно загађење утиче и маса посматраног пакета. Ови коефицијенти се налазе у табели 4, према извору [31].

Коефицијентима ef_t и es_{kt} из претходног модела овде одговарају коефицијенти ef и es_k који не зависе од типа контејнера. Загађење се добија

врста транспорта	коефицијент загађења
брод	8
пруга	22
баржа	31
камион	62

Табела 4: Вредности коефицијената загађења

множењем коефицијената ef и es_t и масе контејнера. При томе маса контејнера обухвата масу пакета и масу празног контејнера (тара). Нови улазни подаци су:

T_t - тара контејнера типа t, где је $t \in [nt]$

ef - коефицијент загађења преко океана

es_k - коефицијенти загађења путовања копном, где је $k \in [nk]$

Осим тога, промељиве f_{ijst} и s_{jkt} су у овом моделу бинарне. Овде се подразумева да се моделом проналази оптимална траса, посебно за сваки тип контејнера, и да се онда користи за све појединачне контејнере тог типа. Затим се за сваки појединачни контејнер посебно рачуна његово загађење.

Два еквивалентна скупа ограничења (3.1) и (3.12) овде се замењују редом са:

$$\sum_{i=1}^{ni}\sum_{j=1}^{nj}\sum_{s=1}^{ns} f_{ijst} = 1, \quad t \in [nt] \tag{3.14}$$

$$\sum_{j=1}^{nj}\sum_{k=1}^{nk} s_{jkt} = 1, \quad t \in [nt] \tag{3.15}$$

Значење ограничења (3.14) је да се за сваки тип контејнера на првом делу пута узима тачно један бродар, тачно једна лука и тачно један сервис. Слично, ограничење (3.15) казује да се за сваки тип контејнера, на другом делу пута, узима тачно једна лука и тачно један вид транспорта до циља. Њихова еквиваленција у моделу описана је ограничењем (3.2).

Функција циља која одговара цени гласи:

$$C(f,s,p) = \sum_{i=1}^{ni}\sum_{j=1}^{nj}\sum_{s=1}^{ns}\sum_{t=1}^{nt}\sum_{l=1}^{nl_t} cf_{ijt}f_{ijst}p_{lt} + \sum_{j=1}^{nj}\sum_{k=1}^{nk}\sum_{t=1}^{nt}\sum_{l=1}^{nl_t} cs_{jkt}s_{jkt}p_{lt} \quad (3.16)$$

Функција циља која одговара загађењу дефинише се на следећи начин:

$$E(f,s,p) = \sum_{i=1}^{ni}\sum_{j=1}^{nj}\sum_{s=1}^{ns}\sum_{t=1}^{nt}\sum_{l=1}^{nl_t} df_{ijs}ef(m_{lt}+T_t)f_{ijst}p_{lt} +$$

$$+ \sum_{j=1}^{nj}\sum_{k=1}^{nk}\sum_{t=1}^{nt}\sum_{l=1}^{nl_t} ds_{jk}es_k(m_{lt}+T_t)s_{jkt}p_{lt} \quad (3.17)$$

3.2 Оптимизација по цени и загађењу, налажење ефикасног и Парето скупа

Вратимо се на модел из тачке 3.1 који је описан ограничењима (3.1) - (3.9). Функције циља које се односе на цену и на загађење дефинисане су редом са (3.10) и (3.11). Задатак који се може разматрати је да се пронађе такав распоред пакета у контејнере и такав избор траса којима се они крећу, да се добије оптимално решење по оба разматрана критеријума (Парето оптимизација). Сваки избор распореда пакета у контејнере и потом избор њихових траса као резултат даје две вредности, укупну цену и укупно загађење за тај избор. Наводимо основне појмове и тврђења у вези са Парето оптимизацијом по два критеријума за рачунање минимума, према приступу у [35, 36, 37].

Сваком избору контејнера и траса x је, на овај начин, придружен уређени пар реалних вредности $(f_1(x), f_2(x))$ где је $f_1(x)$ вредност првог, а $f_2(x)$ вредност другог посматраног критеријума. Критеријуми могу да буду, на пример, цена и загађење, као што ће се овде узети. Скуп свих могућих избора x формира скуп допустивих решења који ћемо означити са X. Критеријумски скуп је $Y = \{(f_1(x), f_2(x))|x \in X\}$.

Дефиниција 3.1. *За уређени пар $(f_1(x), f_2(x))$ кажемо да доминира над уређеним паром $(f_1(y), f_2(y))$ уколико је $f_1(x) \leq f_1(y)$ и $f_2(x) \leq f_2(y)$ и од ове две неједнакости важи бар једна строга неједнакост, тј. $f_i(x) < f_i(y)$ за бар једно $i \in \{1, 2\}$.*

Дефиниција 3.2. *За решење x^* из скупа допустивих решења X каже се да је Парето оптимално уколико не постоји неки $x \in X$, $x \neq x^*$, за који уређени пар $(f_1(x), f_2(x))$ доминира над уређеним паром $(f_1(x^*), f_2(x^*))$. Овакви елементи x^* се називају и недоминирани.*

Дефиниција 3.3. *Скуп свих Парето оптималних решења $x^* \in X$ формира Парето скуп X_{Par}.*

Дефиниција 3.4. *Ако је $x^* \in X_{Par}$ Парето оптимално решење, онда се $y = (f_1(x^*), f_2(x^*))$ зове ефикасна тачка у критеријумском простору Y.*

Дефиниција 3.5. *Скуп свих ефикасних тачака у критеријумском простору $y = (f_1(x^*), f_2(x^*)), x^* \in X_{Par}$ зове се ефикасан скуп и означава са Y_{eff}.*

Ако су скупови могућих вредности $f_1(x)$ и $f_2(x)$ реални интервали и задат је параметар ϵ који одређује максимални размак одређених ефикасних тачака по једној или другој координати, онда се скуп свих ефикасних тачака може одредити следећим алгоритмом. Он се базира на методи ϵ ограничења, а користи и лексикографску методу.

1. корак: Нека је f_1^1 вредност која се добија као минимум функције $f_1(x)$ за $x \in X$. Уколико се тај минимум достиже у више тачака из X, онда се лексикографском методом бира "најмања", тј. она која међу њима има најмању вредност функције f_2, вредност f_2^1. На исти начин се f_2^2 добија као минимум функције $f_2(x)$ за $x \in X$ и, уколико се достиже у више тачака, лексикографском методом се узима најмања међу њима по вредности f_1, f_1^2. Тиме се добијају вредности Парето оптималних маргиналних решења, тачке $Y^1 = (f_1^1, f_2^1)$ и $Y^2 = (f_1^2, f_2^2)$. Уколико је $f_2^1 - f_2^2 < f_1^2 - f_1^1$, онда се по првом критеријуму f_1 врши оптимизација, а по другом f_2 претраживање. У супротном критеријуми f_1 и f_2 замењују улоге. Без смањења општости претпоставимо да је неједнакост $f_2^1 - f_2^2 < f_1^2 - f_1^1$ тачна.

2. корак: Тачке Y^1 и Y^2 се додају у листу ефикасних тачака L, која тиме почиње да се формира. Уколико су вредности функције f_2 цели бројеви нека је $\epsilon = 1$. У противном, ако су на пример вредности функције f_2 рационални бројеви заокружени на две децимале нека је $\epsilon = 0,01$. Нека је $\epsilon_2 = f_2^1 - \epsilon$.

3. корак: Одређује се $\min f_1(x)$ за $x \in X$, уз додатно ограничење $f_2(x) \leq \epsilon_2$. Уколико се добијена минимална вредност f_1^* достиже у више

тачака, онда се лексикографском методом узима она тачка чија је вредност f_2 најмања. Тако добијено решење означимо са $Y^* = (f_1^*, f_2^*)$ и додајмо га у листу ефикасних тачака L.

4. корак: Ако је $f_2^* = f_2^2$, пронађене су све ефикасне тачке, $Y_{eff} = L$. У супротном ставити $\epsilon_2 = f_2^1 - \epsilon$ и вратити се на корак 3.

У 1. и 3. кораку алгоритма одређују се минималне вредности функције описане математичким моделом. За те потребе коришћен је егзактни решавач Lingo.

Пример 3.1. *Одредити ефикасни скуп за инстанцу са вредностима* $m = 200t$ *и* $V = 200m^3$*. Остали улазни подаци узети на основу реалних података који су коришћени и у другим раније поменутим радовима. Узето је да је број доступних контејнера по сваком типу* $nl_t = 9$*, независно од типа* t*.*

Пратећи алгоритам у првом кораку је рачуната само минимална цена, добијена је вредност $12754,40$, док је за такав распоред загађење било једнако $11616,19$. Детаљнијим увидом у добијене резултате видело се да је изабрано 8 малих контејнера и послати су истом, оптималном, трасом.[1] На сличан начин рачунато је минимално загађење и добијена је вредност $11247,96$ при чему су за исте улазне податке добијене три могуће вредности за цену: $12784,00$, $13363,60$ и $13446,40$. У складу са алгоритмом као вредност цене узима се најмања од њих $12784,00$. Тако су добијене прве две тачке ефикасног скупа, оптимална маргинална решења $Y^1 = (f_1^1, f_2^1) = (12754,40; 11616,19)$ и $Y^2 = (f_1^2, f_2^2) = (12784,00; 11247,26)$. Овде је $f_1^2 - f_1^1 < f_2^1 - f_2^2$, па се по f_2 (загађењу) врши оптимизација, а по f_1 (цени) претраживање.

Након тога у складу са 3. кораком алгоритма у ограничења је додато:

$$C \leq C_0 \tag{3.18}$$

где је C вредност функције цене описана са (3.10), а C_0 је константа која се мења изнова при сваком позиву као ϵ_2 у опису алгоритма. Решавањем у Lingu добија се оптимално загађење E које уз остале задовољава и овај услов, а добија се и одговарајућа вредност цене C. Тако добијени

[1] У овом примеру су узети прецизнији улазни подаци па је цена превоза малог контејнера овом трасом $1594,3$, отуда мало одступање у односу на вредности у табели 8 и раду [30] које су заокружене на цео број, па је за мали контејнер оптимална траса имала цену 1594.

парови (C, E) додају се у ефикасан скуп, а за њих добијени распореди пакета у контејнере и контејнера на трасе, елементи су Парето скупа по цени и загађењу.

У табели 5 налазе се резултати налажења ефикасног скупа добијени тестирањем за вредности C_0 које се узимају редом како је описано у 2. кораку алгоритма. Овде су сви улазни подаци, као и резултати који се односе на цену и загађење, задати на две децимале, па се узима да је $\epsilon = 0,01$ и нова вредност C_0 је за толико мања од претходно добијене вредности C. Затим, пошто се овде врши оптимизација по загађењу, а претрага по цени, креће се од тачке Y^2 према Y^1 смањујући C_0. За прву вредност C_0 узима се $12783,99$ и на основу тога се добија нова ефикасна тачка $(C, E) = (12781,80; 11291,53)$. Итеративни поступак се наставља за нову вредност $C_0 = C - 0,01 = 12781,79$. Критеријум заустављања је када се као пар (C, E) добије тачка Y^1. За ову инстанцу улазних података минимална цена је $12754,40$ па зато не постоји решење са мањом ценом. Укупно, са две маргиналне тачке, добијене су 24 тачке које формирају ефикасан скуп, видети табелу 5.

бр.	C_0	C	E	бр	C_0	C	E
1	-	12784,00	11247,26	13	12769,19	12768,50	11474,15
2	12783,99	12781,80	11291,53	14	12768,49	12767,00	11476,00
3	12781,79	12780,30	11293,38	15	12766,99	12765,50	11477,84
4	12780,29	12779,60	11335,80	16	12765,49	12764,80	11520,26
5	12779,59	12778,10	11337,65	17	12764,70	12763,30	11522,11
6	12778,09	12776,60	11339,50	18	12763,29	12761,80	11523,96
7	12776,59	12775,90	11381,92	19	12761,79	12761,10	11566,38
8	12775,89	12774,40	11383,76	20	12761,09	12759,60	11568,23
9	12774,39	12772,90	11385,61	21	12759,59	12758,10	11570,08
10	12772,89	12772,20	11428,03	22	12758,09	12757,40	11612,50
11	12772,19	12770,70	11429,88	23	12757,39	12755,90	11614,34
12	12770,69	12769,20	11431,73	24	12755,89	12754,40	11616,19

Табела 5: Ефикасан скуп добијен у примеру 3.1

У случају да је $f_2^1 - f_2^2 < f_1^2 - f_1^1$ онда се по првом критеријуму f_1 (цена) врши оптимизација, а по другом f_2 (загађење) претраживање. То значи да се уз иста полазна ограничења и функцију циља C, као додатно ограничење уместо (3.18) узима (3.19).

$$E \leq E_0 \tag{3.19}$$

Тада се полази од тачке $Y^1 = (f_1^1, f_2^1) = (12754,40; 11616,19)$, а као прва вредност узима се $E_0 = 11616,18$. Свака следећа вредност E_0 узима се као $E - 0,01$ у односу на одговарајући резултат минимизације цене E. Критеријум заустављања је када се итеративним поступком дође до тачке Y^2. □

Пример 3.2. *Одредити ефикасни скуп за инстанцу са вредностима* $m = 100t$ *и* $V = 500m^3$ *и исте остале улазне податке као у примеру 3.1.*

Код инстанце из примера 3.1 број контејнера ограничава маса, а овде запремина. Сличном методом рачунања узето је да буде $nl_t = 17$ за сваки тип контејнера t. Ово повећава број начина за распоређивање пакета. Почетне ефикасне тачке су: за оптималну цену $(18948,90; 21134,74)$ и за оптимално загађење $(19836,70; 21088,62)$. Да би се одредили ефикасан и Парето скуп поновљен је сличан поступак тражења минималне цене уз додатно ограничење (3.19) које се односи на загађење. Добијани резултати приказани су у табели 6. Може се видети да је на овај начин добијен ефикасан скуп који има само три члана. □

бр.	E_0	C	E
1	-	18948,90	21134,74
2	21134,73	18950,40	21132,89
3	21132,88	18952,60	21088,62

Табела 6: Ефикасан скуп добијен у примеру 3.2

Приликом преласка на наредне три инстанце из табеле 8. уочено је да додавање ограничења облика (3.19) и (3.18) значајно успорава рад. Код последње посматране инстанце за $m = 5000$ и $V = 10000$ број контејнера сваког типа био је 334. Ту и без додатног ограничења, само тражење минималне цене није успео да заврши у року од сат времена. То говори о сложености модела који је овде разматран, са или без додатних ограничења, што чини оптимизацију неефикасном.

Скоро сва имплементирана егзактна и приближна решења у овом раду тестирана су на рачунару са процесором Intel Core i7-2600 CPU 3.40GHz са 12GB RAM на оперативном систему Linux. Изузетак су тестирања урађена помоћу решавача Lingo на рачунару са процесором Intel Core i3-3210 CPU 3.20GHz, са 8GB RAM меморије на 64-битном оперативном систему Windows 8.1.

3.3 Могуће расподеле пакета на контејнере

Постоје три типа стандардних контејнера који се међусобно разликују по димензијама, капацитетима запремине и масе, видети тачку 2.1.1.

У овом делу анализирамо како се задати велики пакет масе m и запремине V може поделити на скуп контејнера три типа тако да је сваки контејнер, осим евентуално последњег додатог, попуњен до лимита своје масе или лимита своје запремине. За тако формиран коначан скуп тројки касније се једноставније бира тројка која је оптимална по задатом критеријуму.

Алгоритам којим је ово решено приказан је својим псеудокодом као Алгоритам 1. Он се своди на одређивање свих могућих тројки ненегативних целих бројева облика (n_1, n_2, n_3) које задовољавају тражене услове, где n_i означава колико има контејнера $i-$тог типа, за $i = 1, 2, 3$. Овде се посебно водило рачуна о редоследу попуњавања контејнера да се не би у решењу могле добити две тројке од којих једна "доминира" над другом, односно садржи другу као свој подскуп. То значи да није могуће добити тројке (n_1^a, n_2^a, n_3^a) и (n_1^b, n_2^b, n_3^b) код којих је $n_1^a \geq n_1^b$, $n_2^a \geq n_2^b$ и $n_3^a \geq n_3^b$, а за бар једну од њих је строга неједнакост тј. $n_i^a > n_i^b$ за неко $i = 1, 2, 3$. Овај део је испрограмиран у програмском језику C.

Алгоритам 1 Алгоритам генерисања уређених тројки броја контејнера

```
procedure TROJKE(V, m, V1, V2, V3, m1, m2, m3)
    S = ∅;
    for i1 = 0 to max{⌈V/V1⌉, ⌈m/m1⌉} do
        for i2 = 0 to max{⌈(V − i1 ∗ V1)/V2⌉, ⌈(m − i1 ∗ m1)/m2⌉} do
            i3 = max{⌈(V − i1V1 − i2V2)/V3⌉, ⌈(m − i1m1 − i2m2)/m3⌉}
            d = 0;
            for (j1, j2, j3) in S do
                if i1 >= j1 and i2 >= j2 and i3 >= j3 then
                    d = 1;
                    break;
            if d == 0 then                    //Ако тројка не доминира постоје
                S = S ∪ {i1, i2, i3};
                print (i1, i2, i3);
```

Алгоритам је ограничен бројем допустивих тројки. Када се фиксирају вредности i_1 и i_2 једнозначно је одређен број контејнера трећег типа (i_3) јер се узима најмањи број контејнера у који може да стане преостала количина. Обилазак у петљама по i_1 и i_2 иде од 0 до највећих потребних вредности за преосталу количину робе. Генерисане тројке се чувају у

скупу S. Овим се постиже да када се дође до нове тројке (i_1, i_2, i_3) може да се провери да ли она доминира неку ранију тројку (j_1, j_2, j_3) из S у ком случају се не додаје у S. Тако су добијене све уређене тројке могућих избора контејнера на које се може распоредити задати пакет.

Пример 3.3. *За задате вредности запремине и масе пакета $V = 200m^3$ и $m = 100t$, генерисати све могуће расподеле овог пакета по контејнерима, тако да се поштују лимити маса и запремина описани табелом 2.*

Имплементирани програм генерисао је укупно 19 могућност и резултати су представљени у табели 7. Ту се види да број потребних контејнера, за све ове могућности, варира од 5 до 7. Овде се може приметити да су тројке $(2, 1, 2)$ или $(6, 0, 1)$ одбачене јер доминирају редом тројке $(2, 0, 2)$ и $(5, 0, 1)$ које су већ од раније у скупу.

У истој табели у две последње колоне приказани су резидуали, масе и запремине које нису попуњене у скупу контејнера који одговара тројци у истом реду табеле. Резидуали се добијају када се од укупних капацитета масе или запремине одузме маса одн. запремина разматраног пакета. □

20' конт. (i_1)	40' конт. (i_2)	40' високи конт. (i_3)	рез. m	рез. V
0	0	5	22,5	150
0	1	4	22,5	140
0	2	3	22,5	130
0	3	2	22,5	120
0	4	1	22,5	110
0	5	0	22,5	100
1	0	4	23,8	110
1	1	3	23,8	100
1	2	2	23,8	90
1	3	1	23,8	80
1	4	0	23,8	70
2	0	2	0,6	0
2	2	1	25,1	50
2	3	0	25,1	40
3	1	1	26,4	20
3	2	0	26,4	10
5	0	1	53,5	20
5	1	0	53,5	10
7	0	0	80,6	10

Табела 7: Број контејнера по типу

Све тројке добијене на овај начин су легитимне (допустиве). Једно

од могућих проширења модела је увођење ограничења у погледу броја присутних контејнера сваког типа, или укупног броја дозвољених контејнера за превоз датог пакета. На пример, у случају представљеном у табели 7, максималан број коришћених контејнера по типовима редом је $(n_1^*, n_2^*, n_3^*) = (7, 5, 5)$. Уколико би присутан број контејнера по типовима био мањи од тога, онда би се неки од набројаних случајева изузели из даљег разматрања.

3.3.1 Најбоље распоређивање пакета

Када се за велики пакет задате масе m и запремине V добију сви могући случајеви његове расподеле по контејнерима алгоритмом 1, онда се може приступити одабиру неког од тих случајева који ће по задатим критеријумима бити оптималан.

Уколико се оптимизација врши само по цени, онда је најпогодније прво урадити оптимизацију по цени за сваки тип контејнера и тако добити њихове оптималне цене и оптималне трасе.

За потребе једноставнијег записа у наставку ћемо користити ознаку $[k] = \{1, ..., k\}$. Ако се оптималне цене транспорта по типу контејнера означе редом са C_1, C_2 и C_3, онда је цена која одговара уређеној тројци (n_1, n_2, n_3) расподеле пакета на контејнере $C = n_1C_1 + n_2C_2 + n_3C_3$. Ако ових генерисаних могућности има k, онда би се оптимизација по цени сводила на тражење $min(\mathcal{C}_i), i \in [k]$, где је $\mathcal{C}_i = n_1^iC_1 + n_2^iC_2 + n_3^iC_3$, а (n_1^i, n_2^i, n_3^i) за $i \in [k]$ генерисана тројка расподеле пакета по контејнерима.

Пример 3.4. *Овакав приступ је коришћен у раду [30]. Ту је разматрана реална ситуација контејнерског транспорта на релацији Шангај-Београд. Добијене су оптималне цене редом по врсти контејнера* $C_1 = 1594EUR$, $C_2 = 2470EUR$, $C_3 = 2483EUR$. *Потом је разматрано пет различитих примера задатих пакета, за сваки од њих су генерисани сви могући распореди пакета по контејнерима, а потом су међу тим могућностима одређене оне које су са минималном ценом. Добијени резултати са оптималним распоредима пакета приказани су у табели 8.*

Оптимизација по загађењу се може урадити на идентичан начин, по истом принципу. Оптимизација по времену је овде најмање дошла до изражаја, јер уколико има довољно контејнера и крену сви у исто време,

$V(m^3)$	$m(t)$	$20DV$	$40DV$	$40HQ$	цена (EUR)
200	200	8	0	0	12752
500	100	1	2	5	18949
1000	500	4	18	0	50836
5000	1000	1	0	71	177887
10000	5000	2	203	0	504598

Табела 8: Оптималне цене за случај пет великих пакета у примеру 3.4

онда ни број контејнера, нити расподела пакета по њима, неће утицати на утрошено време.

Уколико нам није потребан скуп S свих тројки на које се може велики пакет распоредити, одређен алгоритмом 1, него желимо да пронађемо само оптималну тројку на пример по цени, онда се алгоритам може додатно поједноставити и тиме убрзати. Модификовани алгоритам приказан је као Алгоритам 2.

Алгоритам 2 Алгоритам генерисања тројке броја контејнера са оптималном ценом

procedure OptTrojka($V, m, V_1, V_2, V_3, m_1, m_2, m_3, C_1, C_2, C_3$)
  $minC = +\infty$;
  $(mi_1, mi_2, mi_3) = (-1, -1, -1)$;
  for $i_1 = 0$ to $\max\{\lceil V/V_1 \rceil, \lceil m/m_1 \rceil\}$ **do**
    for $i_2 = 0$ to $\max\{\lceil (V - i_1 * V_1)/V_2 \rceil, \lceil (m - i_1 * m_1)/m_2 \rceil\}$ **do**
      $i_3 = \max\{\lceil (V - i_1 V_1 - i_2 V_2)/V_3 \rceil, \lceil (m - i_1 m_1 - i_2 m_2)/m_3 \rceil\}$
      $C = i_1 * C_1 + i_2 * C_2 + i_3 * C_3$;
      if $C < minC$ **then** *//Ако је пронађен нови минимум*
        $minC = C$;
        $(mi_1, mi_2, mi_3) = (i_1, i_2, i_3)$;
  print $(minC, mi_1, mi_2, mi_3)$;

3.4 Модели који се позивају на познате оптималне трасе

У одељку 3.3 описано је како се задати велики пакет може на све могуће начине расподелити на контејнере различитих типова, уз вођење рачуна о ограничењима масе и запремине. Када је позната ова расподела, она се може даље употребити за разне врсте оптимизација, по једном или више параметара. Да би се то применило, потребно је претходно спровести оптимизацију трасе за задате параметре и то по сваком типу контејнера. На пример, када се врши оптимизација по цени, онда се за сваки тип контејнера проналази оптимална траса и добије се минимална цена

превоза која одговара тој траси. У том случају, ако су минималне цене по типу контејнера, редом, C_1, C_2 и C_3, и користи се n_1 контејнера типа 1, n_2 контејнера типа 2 и n_3 контејнера типа 3, онда би цена за превоз целог великог пакета била $C = n_1C_1 + n_2C_2 + n_3C_3$ за ту расподелу пакета на контејнере. О овоме је било речи у одељку 3.3.1.

Међутим, може се запазити да није неопходно да се генеришу све расподеле пакета на контејнере да би се пронашла оптимална расподела и вредност параметара оптимизације. То је илустровано кроз претходна два модела који се не позивају на све генерисане могућности. Они се не ослањају ни на унапред познате оптималне трасе по задатом типу контејнера и параметру оптимизације, него се то код њих рачуна све одједном. Уколико су познате оптималне трасе, може се поједноставити одговарајући математички модел.

Улазни подаци су:

nt - број типова контејнера

nl_t - број контејнера типа t, где је $t \in [nt]$

C_t - минимална цена превоза контејнера задатог типа, где је $t \in [nt]$

V - запремина пакета

m - маса пакета

LV_t - горња граница запремине контејнера типа t, где је $t \in [nt]$

Lm_t - горња граница масе контејнера типа t, где је $t \in [nt]$

Променљиве чијим се варирањем постиже оптимизација су:

p_{lt} - бинарна променљива која говори да ли се превози $l-$ти контејнер типа t, где су $t \in [nt]$, $l \in [nl_t]$

m_{lt} - маса у $l-$том контејнеру типа t, где су $t \in [nt]$, $l \in [nl_t]$

Ограничења која треба да задовољавају променљиве су:

$$\sum_{t=1}^{nt} \sum_{l=1}^{nl_t} m_{lt} = m \tag{3.20}$$

$$m_{lt} \leq Lm_t \cdot p_{lt}, \quad t \in [nt], l \in [nl_t] \tag{3.21}$$

$$\frac{V}{m} m_{lt} \leq LV_t, \quad t \in [nt], l \in [nl_t] \tag{3.22}$$

$$p_{lt} \in \{0, 1\}, \quad t \in [nt], l \in [nl_t] \tag{3.23}$$

$$m_{lt} \geq 0, \quad t \in [nt], l \in [nl_t] \tag{3.24}$$

Вредности промењљивих које задовољавају ограничења треба изабрати тако да се минимизује циљна функција једнака цени:

$$C(p) = \sum_{t=1}^{nt} \sum_{l=1}^{nl_t} C_t p_{lt} \tag{3.25}$$

Слично, уколико се оптимизује загађење, онда се као улазни податак уместо C_t користи E_t као оптимално загађење код превоза контејнера типа t, где су $t \in [nt]$, $l \in [nl_t]$. У том случају функција циља је:

$$E(p) = \sum_{t=1}^{nt} \sum_{l=1}^{nl_t} E_t p_{lt} \tag{3.26}$$

Ограничења представљају подскуп ограничења описаних у одељку 3.1.

Недостатак претходно описаног модела је у томе што су промењљиве одлучивања m_{lt} реалне. На тај начин се умножава број добијених решења која су суштински иста, а разликују се по маси пакета која се налази у неком контејнеру. На пример, ако су два контејнера истог типа попуњена мало више од пола, или је један попуњен до краја а у другом се налази мањи део великог пакета и ако је збир маса исти, онда међу овим решењима нема суштинске разлике. Овај проблем се може превазићи у самом моделу тако што ће се захтевати да сви контејнери истог типа, осим евентуално последњег, буду попуњени до својих лимита масе или запремине. На тај начин се m_{lt} бришу из скупа промењљивих одлучивања и пребацују у улазне податке као m_t јер тада зависе само од типа контејнера. Ове вредности се могу израчунати по формули:

$$m_t = \min \left\{ Lm_t, \frac{LV_t}{V} \cdot m \right\}, \quad t \in [nt]. \tag{3.27}$$

На тај начин као промењљиве одлучивања остају само бинарне p_{lt}. Одговарајућа ограничења постају:

$$\sum_{t=1}^{nt} m_t \Big(\sum_{l=1}^{nl_t} p_{lt} \Big) \geq m \tag{3.28}$$

$$\sum_{t=1}^{nt} m_t \Big(\sum_{l=1}^{nl_t} p_{lt} - 1 \Big) + \epsilon \leq m \tag{3.29}$$

$$p_{lt} \in \{0, 1\}, \quad t \in [nt], l \in [nl_t] \tag{3.30}$$

Ограничење (3.28) казује да уколико би сви узети контејнери били попуњени до максимума, њихова маса треба да буде већа или једнака од масе предвиђеног пакета. Другим речима, то ограничење обезбеђује да изабрани контејнери треба да имају довољну носивост. Пошто је начином рачунања m_t обезбеђено да садржај сваког контејнера задовољава лимите у погледу масе и запремине, тај услов се не наводи у ограничењима. Ограничење (3.29) обезбеђује да, ако би се избацио по један пун контејнер сваког типа, не би била задовољена носивост. Пошто се у моделима не наводе строге неједнакости, овде је додата константа ϵ која се може фиксирати на неку врло малу позитивну вредност, рецимо $\epsilon = 0,0001$. Овим ограничењима сужава се могући избор контејнера управо на скуп коришћен у одељку 3.3. Функција циља остаје непромењена, за цену и загађење су то, редом, функције (3.25) и (3.26).

Разлог зашто се у моделима не користе строге неједнакости је у томе да кодомен треба да буде затворен скуп, јер се тиме се гарантује да се достиже најмања или највећа вредност.

Следећа дорада овог модела се односи на избор улазних параметара nl_t који представљају максимални број контејнера задатог типа. Од nl_t зависи број бинарних променљивих p_{lt} па је потребно изабрати да nl_t има што мању вредност, а да то не спречава да се пронађе оптимално решење. Када се израчуна m_t по формули (3.27), онда се могу на основу тога добити границе за nl_t по формули:

$$nl_t = \left\lceil \frac{m}{m_t} \right\rceil, \quad t \in [nt]. \tag{3.31}$$

Овде $\lceil x \rceil$ означава горњи цео део броја x, тј. цели број k такав да је $k - 1 < x \leq k$.

Уколико се узима да све вредности nl_t у имплементацији модела имају исту вредност, онда се за ту вредност узима њихов максимум. У том

случају је $nl = \max\{nl_t | t \in [nt]\}$.

Овако добијен модел са ограничењима (3.28) и (3.29), као и са дорадом везаном за број доступних контејнера, варијанта је проблема ранца (енгл. knapsack problem). О проблему ранца и разним његовим варијантама може се видети више у [32, 33, 34].

Ова верзија модела се може додатно упростити избацивањем ограничења (3.29) које није суштински значајно јер оптимално решење ће ионако морати да испуни тај услов. У супротном, ако тај услов не би важио, онда би се неки контејнери могли искључити и смањити цена, па решење не би било оптимално. Овако задат модел, са или без ограничења (3.29), Lingo је веома брзо решавао, у делићу секунде чак и за највећи разматрани пакет од $V = 10000m^3$ и $m = 5000t$.

Приметимо да се на основу вредности скупа бинарних променљивих $p_{lt}, t \in [nt], l \in [nl_t]$, добијеног оптималног решења, зна тачно који су контејнери укључени, а који не. Уколико нам то није битно, него само желимо да знамо колико контејнера ког типа чини оптимално решење, онда се то може једноставније одредити на следећи начин: ако су познате све максималне вредности $nl_t, t \in [nt]$, добијене по формули (3.31), онда се у nt угњеждених циклуса или помоу бектрекинг алгоритма могу генерисати све $nt-$орке $(n_1, ..., n_{nt})$, за $n_i \in [nl_i]$, такве да задовољавају "услов довољне носивости":

$$\sum_{t=1}^{nt} m_t \cdot n_t \geq m$$

аналоган услову (3.28). Потом се за оне $nt-$орке, које задовољавају тај услов, одреди вредност функције циља и налази најмања таква вредност, као и одговарајућа $nt-$орка. Уколико је функција циља цена, онда је укупна цена за разматрану $nt-$орку једнака $n_1C_1 + ... + n_{nt}C_{nt}$.

3.5 Анализа и модификација модела

О тематици која је обрађена у овом поглављу било је донекле речи у раду [30]. Као што је поменуто, ту су се генерисале све могућности поделе великог пакета на контејнере у складу са ограничењима масе и запремине сваког типа контејнера. Та ограничења масе и запремине су и овде наведена у формулама (3.4) и (3.5).

У раду [42] разматрана је тематика врло блиска наведеним моделима у претходна три потпоглавља. Ту су узимане у обзир расподеле маса и запремина делова пакета на одабране контејнере који могу да буду различитих типова. Потом су се рачунале оптималне вредности функција циља које представљају цену, односно загађење. Овде је, за разлику од тог рада, раздвојена ситуација на три различита математичка модела ради појашњења, као и због њихових специфичности. Први модел из потпоглавља 3.1 је општији у смислу да дозвољава да се за исти тип контејнера појављује више оптималних путања. Потом се броји колико је контејнера прошло том путањом и на основу тога рачуна укупна цена, односно укупно загађење. Са друге стране, постоји ограничење у моделу узроковано начином рачунања загађења и спречавањем да се за сваки контејнер то загађење рачуна посебно у зависности од његове масе. Зато се у моделу, описаном у потпоглављу 3.1.1, разматрају појединачни контејнери за рачунање загађења и њима одговарајући коефицијенти. Али, у овом моделу се само једна путања по типу контејнера узима за оптималну и она се проналази. Ово је блиско реалној ситуацији – када се пронађе оптимална путања свакако да ће сви контејнери истог типа ићи том путањом. Могао би да се направи модел који је општији од оба описана модела, али би он само додатно закомпликовао приказ ограничења и повећао број променљивих, а не би се суштински добила квалитетнија решења.

Запажамо да се проналажење минималне вредности функција цене по формулама (3.10) или (3.16) може наћи било помоћу егзактних решавача било помоћу имплементације у оквиру неких рачунарских окружења. Овде су коришћена оба начина. Егзактно решавање је урађено помоћу решавача Lingo, а детаљнија имплементација помоћу MATLAB-а. Овде је такође омогућено да, када се пронађу оптималне трасе за сваку врсту контејнера, онда се на основу свих могућих генерисаних распореда пакета на контејнере, рачунају цене превоза за те сценарије. Овде је, такође, могуће добити ранг листу по цени за сваки од ових сценарија. Слична ствар је и за загађење. Потом је могуће, на основу ове две ранг листе, извршити скалирање по најмањим вредностима и сабирање вредности. Овакав приступ је примењен и описан у раду [42].

4 Оптимизација у збирном контејнерском транспорту - проблем паковања

Збирни контејнерски транспорт је врста транспорта у којој се подразумева постојање више пакета које је потребно одједном превести са једне на другу локацију. Претпоставимо да је сваки од њих задат својом масом и запремином. Потребно је да се направи такав распоред пакета по контејнерима којим ће се оптимизовати задате функције циља.

Избор масе и запремине као одлучујућих фактора који описују пакете руковођен је са једне стране праксом која постоји у компанијама које се баве контејнерским транспортом, а са друге стране прописаним лимитима у погледу носивости масе и запремине сваког типа контејнера. Ова ограничења масе и запремине су дефинисана ISO стандардом, а саме вредности ограничења наведене су у табели 1. У пракси је уобичајено да се уместо максималних дозвољених маса и запремина користе нешто мање вредности због тога што се не разматрају тачни геометријски облици пакета. Вредности ограничења масе и запремине које се у пракси више користе, као и овде у наставку, дате су у табели 2. Геометријско паковање пакета је проблем који се бави димензијама самих пакета, али у пракси ни то не мора да буде довољно добро због потребе да се узме у обзир и маса пакета, као и разни технички проблеми везани за тежиште, ломљивост и дозвољене распореде пакета. Ово би све значајно усложнило разматрани проблем. Зато је узета у разматрање његова верзија која обухвата масу, а уместо димензија самих пакета као параметар фигурише њихова запремина, уз смањење укупних лимита масе и запремине. Овде зато нема потпуне гаранције да ће пакети моћи увек да се спакују, јер узимање у обзир масе и запремине јесте неопходан, али не увек и довољан услов. Успешност паковања може да зависи и од других фактора. Зато је ово више применљиво на пакете који имају структуру чији облик није стриктно фиксиран, као код материјала као што су брашно, шећер, со, пиринач, пшеница и слични, чији пакети могу да мењају облик ради лакшег паковања. У том случају се може гарантовати да ако скуп пакета задовољава ограничења масе и запремине, да ће заиста моћи и физички да се спакује у контејнер.

Да би се разматрани проблем паковања у оквиру збирног контејнерског транспорта решио, посебно за велике инстанце када решавачи не

гарантују квалитет добијеног решења, природно је применити метахеуристичке методе. У наставку ће бити примењене методе GRASP и VNS.

4.1 Математички модели и имплементација решења у Lingu

У овом потпоглављу разматра се проблем паковања пакета у контејнере у одређеном месту, при чему су унапред познате цене транспорта сваког типа контејнера од тог места до циља. Ради поједностављења приказа, претпоставимо да је за скуп задатих података претходно пронађена оптимална траса по цени. У том случају је за сваки тип контејнера позната минимална цена његовог транспорта C_t, где је t ознака типа контејнера. Претпоставимо да је задат скуп пакета и да су пакети мањих димензија од контејнера, тако да може више пакета да се спакује у један контејнер. Задатак је да се пронађе оптималан распоред пакета по контејнерима у складу са задатом функцијом циља. Од посматраних параметара и овде су узимани у обзир цена и загађење. Оптимизација по времену у овако постављеном проблему није од значаја, јер уколико постоји довољан број контејнера да се пакети спакују онда сви могу да крену у исто време.

Ради једноставнијег записа модела користиће се следеће ознаке у индексима:

t - редни број типа контејнера,

i - редни број пакета,

Улазни подаци модела су:

nt - број типова контејнера

np - број пакета

C_t - цена превоза контејнера типа t, где је $t \in [nt]$

LV_t - лимит запремине контејнера типа t, где је $t \in [nt]$

Lm_t - лимит масе контејнера типа t, где је $t \in [nt]$

V_i - запремина пакета, где је $i \in [np]$

m_i - маса пакета, где је $i \in [np]$

Ln_t - лимит броја контејнера типа t, где је $t \in [nt]$

Промењљиве одлучивања (енгл. decision variables) овде су:

p_{ijt} - бинарна променљива која одређује да ли се у контејнеру налази пакет, где су $i \in [np]$, $t \in [nt]$, $j \in [Ln_t]$ редни број контејнера типа t

k_{jt} - бинарна променљива која одређује да ли је контејнер узет у обзир, где су $t \in [nt]$, $j \in [Ln_t]$ редни број контејнера типа t

Приликом имплементације тестиране су две варијанте модела које су затим прилагођене решавачу Lingo. На тај начин се међу њима издвојио модел на основу којег овај решавач може да ефикасније генерише решење. Сви модели имају исте претходно описане улазне податке, док је код ограничења било извесних разлика.

Код прве варијанте модела ограничења су:

$$\sum_{t=1}^{nt} \sum_{j=1}^{Ln_t} p_{ijt} = 1, \quad i \in [np] \tag{4.1}$$

$$\sum_{i=1}^{np} p_{ijt} \cdot V_i \leq k_{jt} \cdot LV_t, \quad t \in [nt], j \in [Ln_t] \tag{4.2}$$

$$\sum_{i=1}^{np} p_{ijt} \cdot m_i \leq k_{jt} \cdot Lm_t, \quad t \in [nt], j \in [Ln_t] \tag{4.3}$$

$$p_{ijt} \in \{0, 1\}, \quad i \in [np], t \in [nt], j \in [Ln_t] \tag{4.4}$$

$$k_{jt} \in \{0, 1\}, \quad t \in [nt], j \in [Ln_t] \tag{4.5}$$

Функција циља којом се изражава укупна цена транспорта је:

$$C = \sum_{t=1}^{nt} \sum_{j=1}^{Ln_t} k_{jt} \cdot C_t \tag{4.6}$$

На потпуно сличан начин формулише се модел који се односи на оптимизацију по загађењу. Овде је довољно само на нивоу улазних података C_t заменити са Z_t, оптималним загађењима по сваком типу контејнера. Тада је функција циља, која описује загађење:

$$Z = \sum_{t=1}^{nt} \sum_{j=1}^{Ln_t} k_{jt} \cdot Z_t \tag{4.7}$$

У оба случаја задатак је да се одреди минимум ових функција и тиме се добијају оптимална решења како распоредити пакете по контејнерима.

Значење ових ограничења је следеће: формула (4.1) одређује да се дати пакет може наћи у само једном контејнеру. То је значајно другачији приступ од оног у глави 3 где је подела пакета била могућа. Овде се више пакета може сместити у један контејнер, а помоћу ограничења (4.2) и (4.3) води се рачуна о томе да укупна запремина и маса пакета у контејнеру буде у границима дозвољених лимита запремине и масе. Фактор k_{jt} на њиховој десној страни казује, ако у контејнеру има пакета, онда и тај контејнер мора бити узет у разматрање. Затим, уколико је $k_{jt} = 0$, онда ће и $\sum_{i=1}^{np} p_{ijt}$ бити 0. Случај када у контејнеру нема пакета, није посебно наведен у ограничењима, али ће се то добити израчунавањем функције циља када се тражи минимум. Помоћу (4.4) и (4.5) одређује се да су разматране променљиве p_{ijt} и k_{jt} бинарне.

Када је описани модел искодиран у систему Lingo, он га је класификовао као модел класе PILP (Pure Integer Linear Program). Код тестирања узето је $np = 15$ пакета, случајно су генерисане њихове масе и запремине. Број типова контејнера је $nt = 3$, а број контејнера по сваком типу постављен је такође на 15, колики је и број пакета. Систем је приказао да има 720 променљивих и 106 ограничења од којих су сва била линеарна. За постављени лимит од 5 сати рада Lingo је пронашао као гарантовано најбоље решење вредност 12047 (Objective value) за 2218 секунди (око 37 минута). Вредност функције циља представља цену транспорта у еврима и то ће се подразумевати у наставку. За то време Lingo је извршио скоро 27 милиона итерација.

Анализом ове варијанте модела, уочено је да није у ограничењима наведен услов: ако у контејнеру нема пакета, онда се ни контејнер не рачуна у цену. Дакле, код њега је теоријски могуће да су све променљиве p_{ijt}, за неки фиксни пар j и t, једнаке 0, а да променљива k_{jt} буде једнака 1. Међутим, овакве вредности променљивих не могу да буду оптималне, јер ако се у цену рачунају и контејнери без пакета, онда она не може да буде минимална. За неки пар (j, t) ако је $\sum_{i=1}^{np} p_{ijt} = 0$, тада мора бити $k_{jt} = 0$. Ово се може регулисати ограничењем:

$$\sum_{i=1}^{np} p_{ijt} \geq k_{jt}, \quad t \in [nt], j \in [Ln_t]. \tag{4.8}$$

Додавањем тог ограничења на постојећа добија се варијанта 2 овог модела.

Lingo је и овде одредио исту класу проблема PILP. На истом узорку улазних података од 15 пакета пронашао је да има 720 променљивих и 151 ограничење од којих су сва линеарна. За ограничено време рада од 5 сати пронашао је исто најбоље решење 12047, као и варијанта 1. За то му је било потребно време од 2664 секунде (око 44 минута), гарантовао је да је то најбоље решење и прошао је кроз скоро 30 милиона итерација.

	варијанта 1	варијанта 2
бр. промен.	720	720
ук. огран.	106	151
нелин. огр.	0	0
најбоље решење	12047	12047
време (у s)	2218	2664
итер. (у 10^6)	27	30

Табела 9: Упоредна анализа варијанти решења, за 15 пакета, у Lingu

Перформансе анализираних варијанти модела приказане су у табели 9. На основу ове упоредне анализе, изабрано је да се прва варијанта модела надаље користи за рад решавача са осталим инстанцама.

4.2 Паковање контејнера - категорије и преглед литературе

Проблем паковања пакета у контејнере, описан у претходној тачки, у категоризацији проблема припада паковању бинова (енгл. bin packing).

У свом основном облику проблем паковања бинова се може описати на следећи начин:

Дефиниција 4.1. *Дато је M бинова капацитета 1 и N ставки чије су тежине из интервала $(0, 1]$. Потребно је све ставке спаковати у бинове, тако да се свака ставка налази у тачно једном бину и да укупна тежина ставки у сваком бину буде у границама његовог капацитета. Потребно је одредити најмањи број бинова у које се на овај начин могу спаковати све задате ставке.*

У литератури се проблем паковања бинова често скраћено наводи као BPP, од енглеског назива bin packing problem. Из претходне дефиниције

се може видети да су сви бинови исте величине и да се разматрају само ограничења везана за једну димензију - дужину или тежину бина.

Постоји више врста уопштења овог проблема приликом преласка на две и више димензија. Један од начина је да се задају величине ставки паковања и бинова и да се захтева да збир величина ставки по свакој димензији буде у оквирима величине бина по тој димензији. Овакав начин паковања се у литератури назива векторско паковање (енгл. vector packing). Ово је уопштење дефиниције 4.1 и овде ћемо га исказати у облику математичког модела за случај две димензије.

Слично претходном, нека је дато N ставки које су одређене својим тежинама w_j и v_j као вредностима из $(0, 1]$, и нека је дато M бинова чији су капацитети 1 по свакој од ових димензија. Променљиве одлучивања x_{ij} се односе на то да ли је j-та ставка спакована у i-ти бин или не. Променљиве одлучивања y_i се односе на то да ли је i-ти бин коришћен за паковање или не. Функција циља је $\min \sum_{i=1}^{M} y_i$, где су ограничења:

$$\sum_{i=1}^{M} x_{ij} = 1, \quad j \in [N] \tag{4.9}$$

$$\sum_{j=1}^{N} w_j x_{ij} \leq y_i, \quad i \in [M] \tag{4.10}$$

$$\sum_{j=1}^{N} v_j x_{ij} \leq y_i, \quad i \in [M] \tag{4.11}$$

$$x_{ij} \in \{0, 1\}, \quad i \in [M], j \in [N] \tag{4.12}$$

$$y_i \in \{0, 1\}, \quad i \in [M] \tag{4.13}$$

Значење ових ограничења је такво да се свака ставка пакује у тачно један бин и да по свакој димензији збир тежина бинова не прелази капацитет бина по тој димензији.

Други начин паковања бинова у више димензија има исте улазне податке као и први, али се захтева да се ставке геометријски пакују тако да се одреди просторно решење за њихово паковање у две или три ди-

мензије. Даље подваријанте овог проблема се разликују по томе да ли је дозвољена ротација ставки приликом паковања или није. Овакав вид паковања се у литератури назива паковање кутија (енгл. box packing).

Једна подела у проблему паковања бинова односи се на то да ли су сви бинови једнаки (хомогени проблем) или међу биновима постоје разлике у величини (хетерогени проблем). Хомогени проблем паковања бинова је у литератури значајно више разматран и он је подразумеван, као и у претходним дефиницијама. Приказани модел са ограничењима (4.9)-(4.13) одговара хомогеном векторском паковању, док раније приказани модел са ограничењима (4.1)-(4.5) одговара хетерогеном векторском паковању који ће бити у фокусу овог поглавља.

Код проблема паковања постоји и подела по томе да ли се паковање врши онлајн или офлајн. Приликом онлајн паковања сматра се да се пристиже један по један ајтем (ставка паковања) и да се одмах пакује, а потом прелази на следећи, аналогно игрици Тетрис. Офлајн паковање подразумева да су све ставке паковања познате на почетку, и у литератури је овакав проблем више разматран.

Проблем паковања бинова може се видети и као специјалан случај проблема резања плоча (енгл. Cutting Stock Problem - CSP). Код ове врсте проблема претпоставља се да је за сваки тип пакета задат број примерака. Уколико је број примерака увек један, онда је то специјалан случај - BPP. Детаљнија класификација проблема резања и паковања, уз приказ значајније литературе, дата је у раду [43].

Библиотека која на једном месту прикупља многе инстанце проблема паковања и резања, као и њихова решења и програмске кодове, налази се на интернет адреси `http://or.dei.unibo.it/library/bpplib`. Описана је детаљније у раду [44]. Код проблема паковања ту су највише заступљени једнодимензиони хомогени примери.

Приликом решавања проблема паковања могу се користити егзактне методе или апроксимативни алгоритми. У прегледном раду [45] дат је скорији преглед егзактних метода и значајних радова за проблем паковања бинова. Слично томе у [46] дат је детаљнији преглед апроксимативних алгоритама са богатим прегледом литературе. Оба рада односе се на једнодимензиони проблем паковања бинова.

Овај преглед могућих проблема паковања наведен је из информа-

тивних разлога и ради бољег прегледа позиционирања проблема паковања који се овде решава, а то је векторско хетерогено и хомогено паковање.

4.2.1 Проблем паковања пакета у контејнере као паковање бинова

Проблем паковања контејнера разматран у тачки 4.1, са ограничењима (4.1)-(4.5) и функцијом циља облика (4.6), заправо је варијанта проблема паковања бинова. У питању је дводимензиони проблем векторског паковања бинова, хетерогени, где се паковање врши офлајн.

Овакав тип проблема паковања први пут је уведен и разматран у раду [1] из 1994. године. Као могуће примене проблема аутори наводе придруживање рачунарских процеса процесорима где се процесорско време и меморија узимају као мере, затим придруживање робота радним станицама где су мере време рада и радни простор и, као трећи, складиштење датотека на дискове где су мере везане за време приступа и капацитет складиштења. У том раду су имплементирана три начина решавања проблема, једноставна грамзива хеуристика названа FFOD (енгл. First Fit by Ordered Deviation), симулирано каљење (енгл. Simulated annealing - SA) и генерисање колона (енгл. Column generation - CG). Генерисање колона је егзактна метода којом се у том раду добија решење на основу решавања проблема формулисаног као покривање скупова, потом се за сваки тип бинова покушава генерисање једне колоне ненегативних бројева и у случају неуспеха поступак се наставља методом гранања и ограничавања (енгл. Branch and bound). Решења су тестирана на инстанцама проблема складиштења датотека на четири типа (бинова), а број датотека је био 30, 40, 50, 75 и 100. Посебно се разматра просечан однос капацитета диска и датотека по обе величине, по капацитету меморије и по улазно-излазном оптерећењу, где су узимане у разматрање три могућности: 5, 10 и 20. Разматран је и степен ажурирања датотека, да може да им се повећа величина редом од 1 до 5 процената. Од тих 75 могућности карактеристика инстанце одабрано је 55 и за сваку на случајан начин генерисано по 10 инстанци, укупно 550. Метода FFOD је имала најкраће време рада али и велико одступање решења од најбољег. Метода SA је била спорија са мањим одступањем решења од оптималног. Егзактна решења методом генерисања колона нису рачуната за улазе са више од 50 датотека, а за вредности до 50 она се показала као најспорија.

Следећи рад који се бави нехомогеном проблемом векторског паковања бинова је [47] из 2013. године, објављен 2016. у дорађеној верзији [2]. Овде је разматрана варијанта у којој сваки бин има своје задате димензије. Разматрано је више грамзивих метода заснованих на BFD (енгл. Best Fit Decreasing) и Bin balancing хеуристикама, укључујући разне њихове варијанте, статичке и динамичке по разним ставкама, ајтем-центричне и бин-центричне, три врсте скаларних производа, тако да је укупно разматрано 34 варијанте хеуристика. За тестирање је генерисано 4500 инстанци: било је пет класа улазних података према начину генерисања (случајна равномерна расподела, случајна равномерна расподела са ретким ресурсима, корелисани капацитети, корелисани капацитети и захтеви, слични ајтеми и бинови), за сваку од њих је бирана конфигурација од 10, 30 или 100 бинова, са 2, 5 или 10 димензија и генерисано је по 100 таквих инстанци. Због једноставности алгоритама решења се добијају брзо, али њихов релативни квалитет значајно варира у зависности од изабране варијанте алгоритма и димензије проблема. Овакво паковање је нашло примену код проблема прераспоређивања машина. У другом делу рада је такав приступ примењен на сложенији проблем са додатним ограничењима и са реалним улазним подацима заснованим на Гугловим статистикама (до 5000 машина, 50000 процеса, 12 ресурса).

Разматрани хетерогени проблем векторског паковања пакета у контејнере представља уопштење дводимензионог проблема хомогеног паковања бинова и као такав је NP-тежак. Доказ да је проблем паковања бинова NP-тежак познат је од 1979. године када је у књизи [48] то доказано за хомогени једнодимензиони случај.

Методе којима се може егзактно решити векторски проблем паковања могу бити засноване на гранању и ограничавању (енгл. branch and bound), гранању и одсецању (енгл. branch and cut), генерисању колона. Међутим познате егзактне методе за решавање оваквих проблема нису довољно добре за инстанце веће од неколико стотина објеката. Ово оправдава коришћење метахеуристичких метода за његово решавање.

4.3 Генерисање инстанци

За потребе тестирања реализованих алгоритама за оптимизацију у збирном контејнерском транспорту направљен је генератор случајних величина

пакета. Улазни параметри генератора су број пакета и границе интервала из којих се у складу са униформном расподелом вероватноће бирају масе, односно запремине пакета.

Пример 4.1. *У табели 10 приказане су карактеристике* 10 *пакета, при чему су масе, односно запремине пакета реализације униформно расподељених независних случајних величина из целобројних интервала од 1 до 15 тона, односно од 1 до 25 кубних метара.*

редни број	маса (t)	запремина (m^3)
1	13	1
2	5	8
3	6	23
4	9	9
5	9	3
6	4	21
7	4	24
8	10	24
9	7	17
10	4	1

Табела 10: Пример генерисаних пакета за збирни контејнерски транспорт

За тестирање у систему Lingo остали улазни подаци су $nt = 3$, затим променљиве C_t за $t \in \{1, 2, 3\}$ имају вредности редом $1594EUR$, $2470EUR$, $2483EUR$, као у раду [30]. За лимите масе и запремине узете су стандардне вредности интермодалног транспорта као и у ранијим примерима, наведене су у табели 2. Лимит броја контејнера је постављен на 10 за сваки тип контејнера да не би број контејнера представљао ограничење, него да се симулира случај када контејнера има довољно за оптимални сценарио. Сваки пакет је по димензијама мањи од најмањег контејнера.

контејнер	тип	редни број	пакети
1	1	1	1, 9, 10
2	2	1	2, 3, 5, 6
3	2	2	4, 7, 8

Табела 11: Оптималан распоред пакета по контејнерима

Са овим улазним подацима и наведеном првом варијантом модела, Lingo је за 41 секунду пронашао минималну цену $6534EUR$ за превоз скупа пакета. Решење је приказано у табели 11 и захтева укупно три контејнера: један првог типа $(20DV)$ и два другог типа $(40DV)$. □

Ради бољег поређења резултата метахеуристичких метода и решавача слично претходној инстанци на случајан начин је генерисано више инстанци пакета. Њихове масе и запремине су случајно изабране из истих интервала као у претходном примеру 4.1. Укључујући и инстанцу из тог примера, генерисане су мање инстанце са редом: $10, 11, 12, 13, 15, 20$ пакета, и веће инстанце са $50, 70, 100, 120, 150, 200, 350, 500, 750$ и 1000 пакета. За сваку од већих величина је на случајан начин генерисано по 5 инстанци. Ради прецизности, већим инстанцама додат је индекс, па ће се инстанце од 50 наводити редом као 50_1, 50_2, 50_3, 50_4 и 50_5. Аналогно су означене и остале велике инстанце других величина. Библиотека свих 50 великих и 6 малих инстанци доступна је на адреси `https://doi.org/10.5281/zenodo.5319708`, видети референцу [49].

4.4 Решавање помоћу CPLEX

У наставку је коришћен глобални решавач CPLEX, верзија 12.6.2. и прва варијанта посматраног модела. Број контејнера по сваком типу једнак је броју пакета. Вредности осталих параметара: број типова контејнера, лимити масе и запремине, као и оптималне цене превоза по типу контејнера, исте су као у примеру 4.1. Временско ограничење је постављено на 1 сат и потом на 5 сати, односно, 3600 и 18000 секунди. За то време је једино за мање инстанце, величине 10, 11, 12, 13, 15 и 20 нађено гарантовано оптимално решење. Ови подаци су представљени у табели 12. За све остале инстанце као одговор генерисана су два броја, један који представља нађено минимално решење за то време, и други број који представља границу за коју решавач гарантује да глобални минимум није мањи од тога (Objective bound). Резултати добијени за 10 великих инстанци, где је узимана прва од сваке величине, приказани су у табели 13.

бр. пакета	бр. пром.	ук. огран.	опт. реш.	време (у s)
10	330	70	6534	0,10
11	396	77	7252	0,06
12	468	84	8846	0,06
13	546	91	9722	0,15
15	720	105	12047	0,35
20	1260	140	13786	6,15

Табела 12: Оптимална решења добијена помоћу CPLEX

бр. пак.	бр. пр.	ук. огр.	реш. 1h	гр. 1h	реш. 5h	гр. 5h
50	7650	350	31530	30399,0	31530	30399,0
70	14910	490	48688	45349,2	47970	45349,2
100	30300	700	65898	62834,2	65872	62834,2
120	43560	840	68376	65301,8	68363	65301,8
150	67950	1050	88056	84667,9	88056	84667,9
200	120600	1400	124394	118014,0	121885	118014,0
350	368550	2450	219321	206110,5	218445	206110,5
500	751500	3500	310801	291891,1	308318	291891,1
750	1689750	5250	4910250	439092,3	538794	439092,3
1000	3003000	7000	6547000	591583,6	6547000	591583,6

Табела 13: Добијена решења помоћу CPLEX за 1h и 5h

За свих 10 великих инстанци за сат времена рада CPLEX није могао да гарантује да је добијено оптимално решење. Овде се може запазити да су код највећих инстанци, од 750 и 1000 пакета добијена веома лоша решења, бар десет пута већа од генерисане доње границе. Повећањем времена рада са 1 на 5 сати код 7 инстанци дошло је до поправке добијеног решења, док је код три инстанце, од 50, 150 и 1000 пакета, решење остало непромењено. Границе испод којих се не налази оптимално решење остале су исте за 5 сати као и за 1 сат рада, код свих 10 великих инстанци код којих су решења генерисана.

Закључује се да за веће улазне податке, ни за довољно велико време, на пример 5 сати у табели 13, није могуће гарантовати квалитет решења у CPLEX. Зато постоји потреба за применом метахеуристичких метода за решавање овог проблема.

4.5 Решавање методом GRASP

Метода GRASP (енгл. Greedy randomized adaptive search procedures) је први пут описана у раду [50] из 1989. и потом опширније у раду [51] из 1995. године. Принцип рада ове методе је детаљно приказан у књизи [52], као и у радовима [53, 54]. О бројним применама ове методе може се видети у радовима [55, 56]. Метода GRASP је актуелна и погодна за решавање бројних проблема оптимизације.

Псеудокод примењене метахеуристичке методе GRASP приказан је као Алгоритам 3.

ProblemData се односи на све улазне податке: на списак пакета за-

Алгоритам 3 Предложени GRASP за збирни контејнерски транспорт

```
procedure GRASP(ProblemData, α, t_max)
    n_iter ← 0;
    Kreiraj Sorted list of packages by mass;
    Kreiraj Sorted list of packages by volume;
    repeat
        S ← GRC(Sorted lists of packages, α);  //Полазно случ. изабрано решење
        S' ← LocalSearch(S):                   //Корак локалне претраге
        if f(S') < f(S) then                   //Ажурирање решења
            S ← S';
        n_iter ← n_iter + 1;
    until SessionTime ≥ t_max
```

датих својим масама и запреминама, као и на лимите контејнера по маси и запремини. Вредности α ће у наставку бити детаљније описане.

Имплементација, за потребе овог рада, урађена је у програмском језику C. Суштински се цео поступак израчунавања одвија кроз две велике функције и главну (main) функцију која руководи целим процесом рачунања. Ту је и неколико мањих помоћних функција везаних за копирања података, сортирања и слично.

Решења се представљају једном матрицом чије врсте одговарају контејнерима. У оквиру сваке врсте налазе се редни бројеви пакета који се смештају у тај контејнер. За сваки контејнер се у једном низу памти ког је типа и сваки контејнер узима у обзир границе дозвољених маса и запремина свог типа. Такође, и овде се сваки пакет може сместити у тачно један контејнер. Циљ је да се одреди избор контејнера и распоред пакета у њима, такав да укупна цена буде минимална. Дакле, захтеви су исти као у претходно описаним тачкама, само је начин решавања проблема другачији.

4.5.1 Генерисање полазног решења

GRC (скраћено од енглеског назива Greedy Randomized Construction) је процедура која служи да генерише једно допустиво решење проблема. У облику псеудокода представљена је као Алгоритам 4. Решење се генерише тако што се прво матрица решења иницијализује и попуни нулама. У оквиру једне repeat петље одређује се како се пакети смештају у контејнере, док се сви не сместе. Овде је коришћен приступ којим се код новог контејнера случајним избором прво бира његов тип. Потом се тај контеј-

Алгоритам 4 Greedy randomized construction

```
procedure GRC(Sorted lists of packages, α)
    clm ← Sorted list of packages by mass;
    clv ← Sorted list of packages by volume;
    i ← 0;                          //редни број контејнера
    j ← 0;                          //редни број пакета у контејнеру
    S ← 0;
    b_p ← 0;                        //број тренутно распоређених пакета
    repeat
        if j == 0 then
            Случајно се бира tk(i) тип контејнера i;
        if tk(i) == 1 then
            cl ← clm;
        else
            cl ← clv;
        рачуна c_min на основу cl(1);
        рачуна c_max на основу cl(length(cl)));
        rcl ← ∅;
        for each k ∈ cl do
            if (c(k) ≥ c_min) and (c(k) ≤ c_min + α(c_max − c_min)) then
                rcl ← rcl ∪ {k};
        Случајно се бира k ∈ rcl;
        if пакет k може да стане у контејнер i then
            S(i, j) ← k;
            j ← j + 1;
            b_p ← b_p + 1;
            Брише се пакет k из обе листе clm и clv;
        else
            if j > 0 then                       //Следећи контејнер
                i ← i + 1;
                j ← 0;
    until (b_p ≥ n_p)
    return S;
```

нер попуњава једним по једним пакетом докле год изабрани пакет може да стане у њега. Када то више није случај, прелази се на следећи контејнер и понавља се поступак. Приликом бирања пакета, користи се листа кандидата. Овде је, имајући у виду специфичности самог проблема, одступљено од стандардне верзије процедуре GRC и уместо једне уведене су две листе кандидата. У свакој од њих се налазе сви нераспоређени пакети, али у једној листи су сортирани у опадајућем поретку по маси, а у другој по запремини. Имајући у виду лимите масе и запремине типова контејнера, приказаних у табели 2, ако је контејнер првог типа, коришћена је листа кандидата рангираних по маси, а за контејнере другог и

трећег типа, листа кандидата рангираних по запремини. Када се неки пакет смести у контејнер, он се брише из обе листе кандидата.

Пакет се случајно бира, али не из целе листе, него из њенг подскупа, рестриктивне листе. Претходно се свим члановима листе кандидата додели одговарајућа вредност функције c, па се онда одреди c_{min} и c_{max}. Потом се у рестриктивну листу кандидата узимају само они пакети чија је вредност функције c у интервалу $[c_{min}, c_{min} + \alpha(c_{max} - c_{min})]$. Параметар α је реалан број из $[0, 1]$ и он је један од улазних аргумената функције GRC. Избор функције c је потребно да буде такав да мању вредност имају бољи кандидати, тј. они којима се жели дати приоритет. Како се код контејнера првог типа фаворизују пакети веће масе, онда је у тој листи кандидата по маси за функцију c коришћена реципрочна вредност масе пакета. Слично, код контејнера другог и трећег типа фаворизују се пакети великих запремина па се зато у њиховој листи кандидата за фукнцију c узима реципрочна вредност запремине пакета. На тај начин мања вредност функције c управо фаворизује жељене кандидате.

Уколико неки изабрани пакет не може да стане у текући контејнер, онда се његов избор занемарује и прелази се на следећи, нови, контејнер. Избором новог контејнера и његовог типа, поново се иде на избор пакета из рестриктивне листе кандидата. Овде се води рачуна да се листе кандидата ажурирају избором сваког новог пакета. Осим тога, ажуриране су и вредности c_{min} и c_{max} тако да оне увек одговарају минималној и максималној вредности функције c на тренутној рестриктивној листи кандидата.

Када се распоређивање свих пакета заврши, процедура GRC генерише полазно решење које се враћа алгоритму GRASP. Након тога се позива функција за локалну претрагу, чији је задатак да у околини овог решења потражи локално оптимално решење, односно, такав распоред пакета и контејнера који ће имати минималну цену.

4.5.2 Локална претрага

У литератури постоји велики број различитих функција локалне претраге које подразумевају различите приступе. Од самог проблема који се решава и начина његовог представљања, зависи који приступ и функцију ћемо изабрати.

За проблем паковања који се овде решава карактеристично је да замена места пакетима у оквиру контејнера, као и замена места пакетима у различитим контејнерима, не би дале никакву поправку решења. Наиме, и у том случају укупан број контејнера остаје непромењен. Једино на основу броја контејнера се одређује укупна цена транспорта, а то је функција чији се минимум тражи. Једино потпуно пражњење неког контејнера, пребацивањем свих његових пакета у друге контејнере, могло би да поправи текуће решење. И тај приступ је био окосница локалне претраге. Све док је могуће испразнити неки контејнер, у функцији се покушава то урадити, и стаје се када оваква поправка проверено није могућа.

Код локалне претраге се прво у сваком контејнеру пакети сортирају у растућем поретку по маси. Потом је скуп контејнера, у матрици која представља решење, сортиран у растућем поретку према укупној маси пакета у њима. Алгоритам полази од најмање оптерећеног контејнера и креће од његовог краја, тј. његовог највећег пакета, а затим покушава да га пребаци у последњи контејнер, па ако не успе у претпоследњи и тако редом. Ако успе да га пребаци, онда прелази на остале своје пакете од краја, на исти начин док га не испразни. Ако неки од пакета не успе да пребаци ни у један контејнер, онда се констатује да тај контејнер не може да се испразни, па се поништавају претходна пребацивања његових пакета и поступак се наставља са наредним контејнером. У случају успешног пражњења неког контејнера, добија се решење боље од претходног. Тада се скуп контејнера поново пресортира и поступак креће изнова. Приликом успешног пребацивања појединачног пакета из једног у други контејнер, пресортира се други контејнер, тј. пакети унутар њега. Алгоритам стаје након што покуша да испразни сваки контејнер.

Детаљнији шематски приказ описаног алгоритма дат је као Алгоритам 5. Осим матрице решења S њој се прослеђује и вектор a где се за сваки контејнер чува његов тип. Ознаке које су коришћене имају следеће значење:

- $random(a, b)$ - случајно генерисани цео број из интервала $[a, b]$
- $nBin(S)$ - број контејнера (бинова) у матрици решења S
- $nItem(S, j)$ - број пакета у контејнеру j у матрици решења S

- $Empty(S, j)$ - проверава да ли се испразнио контејнер j у матрици решења S и ако јесте враћа $True$; у супротном враћа $False$.
- $sorted(S, a)$ - сортирање пакета у сваком контејнеру растуће по маси, као и сортирање скупа контејнера растуће према укупној запремини коју носе
- $Transfer(S, j, i, k)$ се односи на процедуру која ако пакет i из контејнера j може да стане у контејнер k, онда се у њега пребацује и враћа $True$; у супротном враћа $False$.

Алгоритам 5 Предложена локална претрага за GRASP

```
procedure LocalSearch(S, a)
    if (random(0, 1) = 0) then
        (S, a) ← ImproveByType(S, a);
    (S', a') ← sorted(S, a);
    for (j ← 1; j ≤ nBin(S'); j++) do
        for (i ← nItem(S', j); i ≥ 1; i−−) do
            for (k ← nBin(S'); k ≥ 1; k−−) do
                if (Transfer(S', j, i, k)) then
                    break;
        if (EmptyBin(S', j)) then
            (S', a') ← sorted(S', a');
            (S, a) ← (S', a');
        else
            (S', a') ← (S, a);
    (S', a') ← ImproveByType(S', a');
```

Овде је код сортирања вођено рачуна да то буде по маси јер су капацитети маса свих типова контејнера приближно исти. Приликом пребацивања појединачних пакета води се рачуна о ограничењима масе и запремине, тј. узиман је у обзир и тип контејнера. На крају локалне претраге додата је провера да ли се у постојећој расподели пакета по контејнерима сваком појединачном контејнеру може променити тип тако да задржи постојеће пакете, поштујући ограничења масе и запремине, а да му се променом типа смањи вредност функције циља. Ово се показало као делотворно па је и после фазе размрдавања, на почетку локалне претраге, уведена иста провера којом се покушава смањити тип контејнера али која се не извршава на почетку сваке локалне претраге него се о томе одлучује са вероватноћом од 50%.

Уколико локалном претрагом није пронађено боље решење, задржава се оно решење које је добијено процедуром GRC. Тиме је завршена једна

итерација. Потом се прелази на нову итерацију и тако док се не испуни критеријум заустављања. Решење после сваке итерације се упоређује са до тада најбољим, и ако је боље, задржава се.

4.5.3 Имплементиране варијанте и тестирање њихових параметара

Избор параметра α је критеријум на основу којег се разликују разне варијанте методе GRASP. Овде су имплементиране три варијанте методе GRASP, назовимо их редом B-GRASP, U-GRASP и R-GRASP.

Метода B-GRASP је основна (енгл. basic). Код ове методе параметар α, који одређује дужину листе кандидата при генерисању полазног решења, фиксиран је на константну вредност из интервала $(0, 1]$. Приликом имплементације за α су тестиране вредности из скупа

$$\alpha \in \{0,05; 0,1; 0,15; 0,2; 0,25; 0,3; 0,45; 0,4; 0,45; 0.5\}.$$

Приликом тестирања параметара коришћено је време рада од 20 секунди, за сваку одабрану инстанцу метода се извршавала 10 пута и приказано је најбоље добијено решење. Тестирања су извршена на скупу од 10 великих инстанци, првих иы сваке групе од 5 инстанци исте величине. Приликом поређења као критеријум квалитета разматран је највећи број постигнутих најбољих решења за ту вредност параметра. Код методе B-GRASP показало се да се за мање вредности α добијају бољи резултати. У том случају је листа кандидата мања и фокусирана је на боље кандидате. Резултати тестирања су приказани у табели 14. На крају, међу свим испитаним кандидатима за α као најбоља се показала вредност $0,05$.

Метода U-GRASP подразумева да се α равномерно (униформно) бира из задатог интервала облика $[0, 1/k]$ где је k неки природан број. Случајно бирање α понавља се сваки пут кад се генерише ново полазно решење. Овде су за k тестиране све вредности из скупа од 1 до 10. Резултати тестирања су приказани у табели 15. Најбоље перформансе је показала варијанта у којој је $k = 10$. То значи да се α сваки пут равномерно бира из интервала $[0; 0, 1]$.

Метода R-GRASP је реактивна метода у којој се вредности параметра α мењају кроз итерације. Оваква варијанта GRASP методе први пут је описана у раду [57] из 2000. године. Наиме, полази се од фиксног скупа D_α могућих вредности за α. Овде је узето да то буде скуп од $m = 5$

бп.	0,05	0,1	0,15	0,2	0,25
50	**32380**	32419	33111	33098	32406
70	49564	**48859**	49577	50401	50401
100	**67637**	68487	69231	69324	69337
120	**70939**	71696	71657	71683	71657
150	**93050**	93156	93063	93900	93063
200	**130331**	131181	131194	132083	130305
350	**226701**	227551	227432	229013	228124
500	**321581**	322301	323229	323934	324626
750	482666	**479880**	484965	486157	487653
1000	**650107**	651260	653427	656001	656747
бп.	0,3	0,35	0,4	0,45	0,5
50	33230	33243	33243	33243	33243
70	49709	49735	49748	50414	49735
100	69337	69363	69363	69218	69311
120	72388	71657	71657	71670	71644
150	93913	93913	93900	94605	94644
200	131912	131767	131220	131912	131596
350	228124	228816	228005	229612	228974
500	324955	324704	323921	325357	323763
750	489695	487383	489866	488910	487982
1000	658992	657004	657517	658808	656444

Табела 14: Тестирање параметра α за методу B-GRASP за 20s

различитих иницијалних вредности

$$\alpha \in \{0,05; 0,1; 0,15; 0,2; 0,25\} = D_\alpha.$$

Свакој од њих се у почетку додељује подједнака вероватноћа избора, $p_i = \frac{1}{m}$, $i \in [m]$, тј. $p_i = 0,2$ у овом случају. Потом се у свакој итерацији α бира на случајан начин из полазног скупа, са вероватноћом избора сваке вредности p_i. За сваку коришћену вредност α_i рачуна се просечна вредност добијене функције циља A_i у свакој итерацији. На основу тога се прерачунавају вероватноће p_i, где је

$$p_i = \frac{q_i}{\sum_{j=1}^{m} q_j}, \quad q_i = \frac{z^*}{A_i}.$$

Овде је са z^* означена до тада најбоља пронађена вредност функције циља. Ова модификација вредности p_i повећава вероватноће за оне α_i за које су се добијала боља решења.

Како је за ажурирање p_i потребно користити A_i, онда се у првих $m = 5$

бп.	1	2	3	4	5
50	**32380**	32393	32393	**32380**	**32380**
70	48885	48714	48859	48859	48714
100	68342	67492	67637	67637	67466
120	70952	70833	70794	70781	70102
150	92397	92332	92358	92944	91363
200	130904	129481	129507	129468	129494
350	227077	225496	226214	225403	225719
500	319474	317880	319461	319487	**317696**
750	480150	478478	479577	479549	478893
1000	648125	647446	645510	645733	644541
бп.	6	7	8	9	10
50	**32380**	**32380**	**32380**	**32380**	**32380**
70	48846	48714	48714	48022	**47996**
100	67611	67598	66787	67479	**66774**
120	70794	70807	**70089**	70781	70794
150	**91350**	92319	92226	91534	91534
200	129297	129468	**128802**	129455	129468
350	225496	**223757**	225390	225470	225548
500	317854	317893	318092	318717	317841
750	479940	479940	479787	479142	**477587**
1000	644963	**643263**	643336	644766	643540

Табела 15: Тестирање параметра k за методу U-GRASP за 20s

итерација за α користе, редом, вредности α_i да би све A_i добиле иницијалне вредности. Након тога да се не би превише често прерачунавале вредности p_i, може се то радити после одређеног броја итерација γ. За γ су тестиране вредности из скупа од 100 до 1000 са кораком 100, од којих се 500 показала као најбоља. Овде је зато изабрано да се ажурирање ради на сваких $\gamma = 500$ итерација. Резултати тестирања су приказани у табели 16. Псеудокод Алгоритам 6 одговара методи R-GRASP. Ту су у односу на Алгоритам 3 присутни кораци везани за иницијализовање и ажурирање вероватноћа избора α, као и улазни параметри γ и D_α уместо α, што остале варијанте методе GRASP не користе.

Помоћу α се регулише дужина рестриктивне листе кандидата у процедури GRC. Што је α мање, онда се листа сужава на фаворите и тиме се добија нешто боље решење. Након извршених тестирања, уочено је да се вредности вероватноћа p_i не мењају значајно. Главна промена је била у томе да $\alpha_1 = 0,05$ добија нешто већу вероватноћу, а највеће вредности α добијају временом нешто мање вероватноће. Тако је, на пример, за случај 50 пакета и време од 120 секунди, у сваком од 30 извршења било

бп.	100	200	300	400	500
50	**32393**	**32393**	**32393**	**32393**	**32393**
70	**48859**	**48859**	**48859**	**48859**	**48859**
100	**68381**	**68381**	**68381**	**68381**	**68381**
120	**70965**	**70965**	**70965**	**70965**	**70965**
150	93887	93887	**93247**	93887	**93247**
200	**130489**	**130489**	**130489**	**130489**	**130489**
350	228176	228176	228308	228163	**227551**
500	322537	323281	323281	323281	**322340**
750	483760	482936	**482257**	483042	**482257**
1000	651180	650382	651180	**650001**	**650001**
бп.	600	700	800	900	1000
50	**32393**	**32393**	**32393**	**32393**	**32393**
70	**48859**	**48859**	**48859**	**48859**	**48859**
100	**68381**	**68381**	**68381**	**68381**	**68381**
120	**70965**	**70965**	**70965**	**70965**	**70965**
150	**93247**	**93247**	93887	**93247**	93887
200	**130489**	**130489**	**130489**	**130489**	**130489**
350	228308	227966	228308	227966	227966
500	**322340**	322537	**322340**	**322340**	**322340**
750	**482257**	**482257**	**482257**	483226	482482
1000	**650001**	650382	650382	650931	651180

Табела 16: Тестирање параметра γ за методу R-GRASP за 20s

Алгоритам 6 Предложени R-GRASP за збирни контејнерски транспорт

procedure GRASP($ProblemData, D_\alpha, \gamma, t_{max}$)
  $n_{iter} \leftarrow 0$;
  Иницијализуј вероватноће избора $\alpha \in D_\alpha$;
  Креирај *Sorted list of packages by mass*;
  Креирај *Sorted list of packages by volume*;
  repeat
    Изабери вредност $\alpha \in D_\alpha$ за текућу итерацију;
    $S \leftarrow GRC(Sorted\ lists\ of\ packages, \alpha)$; *//Полазно случ. изабрано решење*
    $S' \leftarrow LocalSearch(S)$: *//Корак локалне претраге*
    if $f(S') < f(S)$ **then** *//Ажурирање решења*
      $S \leftarrow S'$;
    $n_{iter} \leftarrow n_{iter} + 1$;
    if $n_{iter} mod \gamma = 0$ **then**
      Ажурирај вероватноће избора $\alpha \in D_\alpha$;
  until $SessionTime \geq t_{max}$

између 1925 и 1936 хиљада итерација. Код сваког извршења на крају је p_1 једнако $0,2047$, а p_5 узима вредности из интервала $0,1985 - 0,1986$, што су и највећа одступања од иницијалних вероватноћа $0,2$.

У погледу критеријума заустављања, коришћено је временско ограни-

чење. Због фактора случајности присутног у метахеуристичким методама, како би се добили поузданији показатељи квалитета који могу и статистички да се обраде, за исте параметре коришћено је по 30 извршења програма. На пример, ако се за неку инстанцу постави да је временска граница 120 секунди, то се односи на једно извршење, а програм се покреће 30 пута како би се добили усредњени резултати.

Све три варијанте методе GRASP и њима добијени резултати представљени су у раду [5] из 2019. године.

4.5.4 Добијени резултати

Методе B-GRASP, U-GRASP и R-GRASP тестиране су прво на малим инстанцама од редом: 10, 11, 12, 13, 15 и 20 пакета. Коришћене су исте инстанце које су тестиране и применом решавача CPLEX, ради лакшег поређења добијених резултата. Приликом тестирања уз временско ограничење од 120 секунди, у свих 30 извршења, програм је генерисао сваки пут исто решење. Добијена су иста решења која је као оптимална добио CPLEX и која су приказана у табели 12.

Резултати за методу U-GRASP на скупу малих инстанци представљени су у табели 17. За називе колона коришћени су скраћене ознаке:

- *најр* - најбоље решење,
- *пвдн* - просечно време до налажења најбољег решења,
- *пуби* - просечан укупан број извршених итерација за задато време изражен у милионима,
- *пидн* - просечан број итерација који је потребан да се добије најбоље решење.

бр. пакета	*најр*	*пвдн*	*пуби*	*пидн*
10	6534	0,0	21,965	10,97
11	7252	0,0	19,588	4,27
12	8846	0,0	16,657	1,47
13	9722	0,0	13,246	12,1
15	12047	0,003	11,503	302,8
20	13786	0,019	9,064	1444

Табела 17: Решења добијена методом U-GRASP за 120s

Како су се поклопила најбоља добијена решења на овим мањим инстанцама са решењима добијеним помоћу CPLEX, може се закључити да је оправдано коришћење метода и на већим инстанцама. Код већих инстанци већа и разноликост добијених најбољих решења у $n = 30$ извршења, па постоји потреба за праћењем још неких параметара који мере та одступања пронађених решења од најбољег решења у свих 30 извршења.

Једна од коришћених мера варијације података је просечно одступање (енгл. average gap). Оно се рачуна помоћу израза:

$$agap = \frac{100}{T^* \cdot n} \cdot \sum_{i=1}^{n} (T_i - T^*). \tag{4.14}$$

Овде T_i представља најбоље пронађено решење у $i-$том извршењу програма. T^* је најбоље пронађено решење у свих n извршења. Како се резултат множи са 100, добијена вредност је у процентима.

Друга коришћена мера је стандардна девијација у односу на најбоље пронађено решење, такође изражена у процентима и рачуната помоћу израза:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n} (T_i - T^*)^2}{n}} \cdot \frac{100}{T^*}. \tag{4.15}$$

Обе ове мере у вези су са стабилношћу методе при генерисању најбољег решења кроз 30 покретања и што су ближе 0 сматра се да је добијен квалитетнији резултат.

Све три варијанте методе GRASP, тестиране су на скупу од 10 великих инстанци од редом 50, 70, 100, 120, 150, 200, 350, 500, 750 и 1000 пакета. Време рада било је ограничено на 120 секунди по извршењу. У табелама 18, 19, 20 и 21 приказани су резултати добијени тестирањем све три варијанте методе GRASP на скупу од 10 великих инстанци. У табели 18 дат је упоредни приказ добијених решења помоћу решавача CPLEX за 1 сат рада, и GRASP за 120 секунди рада по једном извршењу. Решења до којих је CPLEX дошао нису оптимална, он је осим ових решења рачунао и доњу границу оптималног решења и његов рад је заустављен због временског ограничења, а не због проналаска оптималног решења. У табели 18 се види да је метода U-GRASP на свих 10 инстанци добила боље или једнако решење од преостале две варијанте GRASP, па се тако издвојила као најбоља. У поређењу са резултатима

CPLEX за 1 сат рада U-GRASP је добила код 2 инстанце боља решења (750 и 1000 пакета), док је код 8 инстанци добила лошија решења (50-500 пакета).

бр. пак.	CPLEX 1h	B-GRASP	U-GRASP	R-GRASP
50	**31530**	32380	32367	32380
70	**48688**	48714	47996	48846
100	**65898**	67598	66774	67611
120	**68376**	70076	70076	70102
150	**88056**	91534	91482	92332
200	**124394**	129468	128631	129455
350	**219321**	225641	224475	225654
500	**310801**	319909	317149	320772
750	4910250	480940	**477587**	480901
1000	6547000	645087	**641267**	647215

Табела 18: Решења добијена помоћу CPLEX за 1h и GRASP за 120s

бр. пак.	B-GRASP	U-GRASP	R-GRASP
50	**36,345**	49,647	44,637
70	54,359	**44,123**	57,249
100	54,203	55,912	**43,151**
120	60,523	67,834	**56,872**
150	70,484	65,109	**64,044**
200	65,716	**48,729**	65,746
350	**59,957**	62,560	70,182
500	**51,992**	54,276	60,897
750	70,104	**60,861**	66,537
1000	59,101	**50,823**	57,540

Табела 19: Просечно време (s) до најбољег решења методом GRASP

У табели 19 упоредо је приказано просечно време потребно да се добије најбоље решење за све три представљене варијанте методе GRASP. Овде је ситуација уједначена, само је варијанта U-GRASP нешто боља од осталих, јер је код 4 од 10 тестираних инстанци имала најмању вредност просечног времена.

У табели 20 приказан је просечан укупан број итерација и просечан број итерација до најбољег решења за све три варијанте методе GRASP. Код укупног броја итерација нема значајне разлике међу приказаним варијантама, док је код броја итерација до најбољег решења нешто боља варијанта U-GRASP од осталих, јер је ова особина сразмерна са временом потребним да се пронађе најбоље решење. Број итерација опада са величином инстанци јер је временска граница иста.

бр. пак.	B-GRASP	U-GRASP	R-GRASP	B-GRASP	U-GRASP	R-GRASP
50	1950192	1959490	1932656	590658	810498	719004
70	1158034	1121531	1107575	524579	412183	528335
100	638524	609191	601519	288410	283811	216300
120	511539	477307	476103	257992	269854	225632
150	331300	318314	312765	194592	172711	166948
200	190803	189132	176639	104488	76796	96769
350	65214	64337	60287	32583	33540	35295
500	33700	33210	31142	14600	15020	15803
750	15223	15058	14283	8891,6	7635,0	7918,8
1000	8481,2	8385,5	7977,3	4175,3	3549,8	3823,6

Табела 20: Просечан укупан број итерација и просечан број итерација до најбољег решења помоћу GRASP

бр. пак.	B-GRASP	U-GRASP	R-GRASP	B-GRASP	U-GRASP	R-GRASP
50	**0,020**	0,048	0,046	**0,028**	0,052	0,054
70	**0,630**	1,513	0,979	**0,883**	1,572	1,208
100	**0,192**	1,093	0,774	**0,423**	1,134	0,959
120	1,448	**0,432**	1,863	1,533	**0,661**	1,966
150	1,527	**0,868**	0,903	1,599	**0,949**	1,039
200	**0,637**	0,691	1,080	0,737	**0,732**	1,145
350	0,591	**0,405**	0,876	0,654	**0,464**	0,918
500	0,625	0,525	**0,518**	0,668	0,563	**0,562**
750	**0,346**	0,388	0,505	**0,370**	0,415	0,538
1000	0,618	**0,495**	0,548	0,651	**0,526**	0,581

Табела 21: Добијене вредности $\bar{a}$ и σ помоћу GRASP

У табели 21 могу се видети добијене вредности статистичких параметара $\bar{a}$ и σ. Овде је метода B-GRASP нешто боља код $\bar{a}$, а U-GRASP нешто боља код σ. Мале вредности ових параметара значе да су добијена решења међусобно блиска.

Имајући у виду све приказане особине варијанти методе GRASP, пре свега најбоље добијене вредности функције циља, може се сматрати да је варијанта U-GRASP боља од осталих, па ће се само она у наставку и разматрати. Ради међусобног поређења у табели 22 приказани су резултати ове методе за 120 секунди рада на скупу од 10 великих инстанци. За називе колона, које се поклапају са описима података у претходним табелама, коришћени су скраћене ознаке:

- *најр* - најбоље решење,
- *прдн* - просечно време до најбољег,

- *пуби* - просечан укупан број итерација,
- *пидн* - просечан број итерација до најбољег решења.

бр. пак.	*најр*	*пвдн*	*пуби*	*пидн*	agap	σ
50	32367	49,647	1959490	810498	0,048	0,052
70	47996	44,123	1121531	412183	1,513	1,572
100	66774	55,912	609191	283811	1,093	1,134
120	70076	67,834	477307	269854	0,432	0,661
150	91482	65,109	318314	172711	0,868	0,949
200	128631	48,729	189132	76796	0,691	0,732
350	224475	62,560	64337	33540	0,405	0,464
500	317149	54,276	33210	15020	0,525	0,563
750	477587	60,861	15058	7635,0	0,388	0,415
1000	641267	50,823	8385,5	3549,8	0,495	0,526

Табела 22: Добијена решења помоћу U-GRASP за 120s

Да би се метода U-GRASP упоредила са резултатима добијеним помоћу CPLEX, она је осим на полазном скупу од 10 великих инстанци тестирана и на још 40 великих инстанци. Код ових тестирања коришћено је време од 120 секунди рада по извршењу. Код CPLEX је коришћено време од 1 сата по извршењу. Резултати тестирања приказани су у табели 23. Решења до којих је дошао CPLEX ни овде нису гарантовано оптимална. Може се уочити да је у задатим временским оквирима на ових 40 инстанци код њих 12 U-GRASP добио боље решење, код 1 инстанце су добили исто решење и на осталих 27 инстанци је CPLEX добио боље решење. На мањим инстанцама боља решења је постигао CPLEX, а на већим U-GRASP.

Овде је потребно истаћи различиту природу хеуристичких метода, као што је GRASP и егзактних решавача као што је CPLEX. Хеуристичке методе не могу да гарантују оптималност решења, без обзира колико време рада имају на располагању. Егзактни решавачи, када дођу до оптималног решења, они гарантују за њега. Када не могу да пронађу оптимално решење они испоручују најбоље решење до којег дођу, али и границу за коју гарантују да оптимално решење не може да буде боље од тога.

Може се закључити да се имплементирана GRASP метода показала као релативно квалитетна у решавању проблема векторског паковања, овде у виду оптимизације у збирном контејнерском транспорту. Она

инст.	CPLEX	U-GRASP	инст.	CPLEX	U-GRASP
50_2	**31346**	31359	200_2	**121543**	127361
50_3	**32577**	**32577**	200_3	**127214**	132366
50_4	**27598**	27611	200_4	**117313**	119601
50_5	28526	**28513**	200_5	**125436**	129580
70_2	**44650**	45487	350_2	229292	**228762**
70_3	**43564**	44388	350_3	**214425**	218796
70_4	**42833**	43696	350_4	**220433**	223236
70_5	**43919**	44637	350_5	**222172**	224257
100_2	**61453**	62303	500_2	324626	**320627**
100_3	**57936**	59610	500_3	**301053**	307855
100_4	**63778**	65491	500_4	353012	**335012**
100_5	**63581**	66077	500_5	**308673**	314376
120_2	**78492**	81055	750_2	4910250	**479827**
120_3	**76167**	77201	750_3	4910250	**481910**
120_4	**72256**	73982	750_4	4910250	**481446**
120_5	**73487**	75918	750_5	4910250	**488107**
150_2	**93898**	97231	1000_2	6547000	**631597**
150_3	**89274**	90137	1000_3	6547000	**651655**
150_4	**93219**	95624	1000_4	6547000	**645445**
150_5	**88945**	91324	1000_5	6547000	**649685**

Табела 23: Добијена решења помоћу CPLEX за 1h и U-GRASP за 120s

за мање инстанце даје оптимална решења као и CPLEX, док код већих инстанци показује своју предност тако што код неких инстанци за битно краће време проналази боља решења него CPLEX за 1 сат рада. Такође, овом методом је омогућено да се реше проблеми које CPLEX није успео да реши тачно због величине инстанци. Ипак код мањих у групи великих инстанци метода U-GRASP није успела да добије решење боље од CPLEX. Ово је било мотивација да се ради на развоју још једне метахеуристичке методе за решавање истог проблема – у питању је метода VNS која је описана у наставку.

4.6 Решавање методом VNS

Метода променљивих околина позната као VNS, што је скраћено од енглеског назива Variable Neighborhood Search, убраја се у новије оптимизационе поступке који су се показали као врло ефикасни приликом решавања многих класа проблема оптимизације. Методу VNS су увели Младеновић и Хансен 1997. године у раду [58]. О самој методи VNS, осим у поменутом раду, може се прочитати више у радовима [59, 60, 61].

Овде су примењене варијанте методе VNS које имају све кључне елементе основне методе VNS. Основни псеудокод је приказан Алгоритмом 7.

Алгоритам 7 Основна структура методе VNS

```
procedure VNS(Problem Data, r_max, t_max)
    Генерисање полазног решења S;
    repeat
        r ← 1;
        while r ≤ r_max do
            S' ← Shake(S, r);              //Фаза размрдавања
            S'' ← Local Search(S');        //Фаза локалне претраге
            if f(S'') < f(S) then          //Да ли се решење замењује или не
                S ← S'';
                r ← 1;
            else
                r ← r + 1;
    until SessionTime ≥ t_max
```

4.6.1 Генерисање полазног решења

Матрични начин представљања решења, као и друге променљиве, врло су слични као у претходно описаној методи GRASP. Оно што је овде другачије је постојање засебне функције за иницијализацију решења која се позива само једном на почетку рада програма, а не за сваку итерацију. Та функција је овде осмишљена тако да се у њој користи само најмањи тип контејнера. Пакети се у почетку сортирају у опадајућем поретку лексикографски по маси и запремини, па се тим редом један по један слажу у контејнере. Пакет који се разматра покушава се спаковати редом у постојеће контејнере, ако не може да стане ни у један од њих почиње се нови контејнер. Може се приметити да је ова функција једноставнија у поређењу са процедуром GRC код методе GRASP, јер нема могућности случајног избора типа контејнера. Такође, нема могућност случајног избора пакета из рестриктивне листе кандидата где су код малих контејнера приоритет имали нераспоређени пакети највеће масе. Оно што им јесте заједничко то је приступ да, код малих контејнера, пакети највеће масе имају приоритет у распоређивању. Текуће решење се, слично методи GRASP, чува у облику матрице у којој један ред одговара једном контејнеру. У том реду се један за другим налазе индекси пакета смештених тако да се поштују укупна ограничења масе и

запремине за сваки контејнер, у зависности од његовог типа. У посебном низу се памте типови свих контејнера.

4.6.2 Размрдавање и локална претрага

Друга значајна разлика метода VNS и GRASP је у такозваној фази размрдавања или шејкинг фази (на енглеском shaking) која је специфичност методе VNS. У тој фази се до тада најбоље решење, у почетку је то иницијално, измени. Као што се у псеудокоду може видети, дефинише се променљива red, која узима вредности од 1 до rmax. Ако је red, на пример 5, онда се 5 пута извршава део који модификује тренутно решење. Овде су изабрана три начина модификације који се реализују један за другим, као три врсте дозвољених потеза. Први потез подразумева да се из текућег решења случајно изабере контејнер и да му се на случајан начин изабере нови тип. Та промена се прихвата уколико је могућа, при чему се води рачуна о тренутном саставу пакета тог контејнера, као и о лимитима маса и запремина његовог типа. Ако промена типа није могућа, онда нема никаквих дешавања. Други начин измене решења заснива се на случајном избору два различита контејнера и у њима се потом случајно бирају два пакета који замене места, уколико је то могуће. И овде се води рачуна о лимитима масе и запремине типова контејнера као основном услову одобравања замене. Приметимо да се захтева да изабрани контејнери буду различити, јер ако се замењују пакети у оквиру истог контејнера, онда то нема утицаја - добија се исти распоред пакета по контејнерима. Код овог потеза је могуће додати нови контејнер са једнаком вероватноћом као што је вероватноћа избора сваког постојећег контејнера. Такође, омогућено је да се приликом избора пакета осим постојећих узме и 0. У том случају разматрани контејнер не даје пакет него само добија пакет који се пребацује из другог контејнера. Трећи потез подразумева случајан избор контејнера и покушај његовог пражњења тако што се сви пакети пребацују у остале контејнере. Уколико се при томе дође до пакета који не може да се пребаци у постојеће контејнере, поступак за тренутно разматрани контејнер стаје, он остаје евентуално делимично пребачен. Прва два потеза се понављају одређен број пута. Параметар који одређује број понављања ова два потеза је ред. После њих се понавља и трећи потез исти број пута. Ред, као број извршавања потеза у фази размрдавања расте

од 1 до границе rmax. Када се за фиксирани ред, на пример 5, толико пута (5) изврши једна па друга петља, које обављају три описана начина замене у сваком проласку, тиме је завршена једна фаза размрдавања.

Након размрдавања, уколико је дошло до промене инстанце, позива се функција за локалну претрагу. Ова функција је урађена на сличан начин као у описаној методи GRASP. Покушавају се испразнити најмање оптерећени контејнери пребацивањем пакета у друге контејнере. Приметимо да је пражњење контејнера једини начин да дође до смањивања укупне цене транспорта. Фаза локалне претраге користи околину која је описана само трећим потезом поменутим у фази размрдавања, уз одређене измене, јер се контејнер за пребацивање не бира случајно. Контејнери се сортирају растуће по укупној запремини. У сваком контејнеру се пакети сортирају растуће лексикографски по запремини и маси. Из таквог низа контејнера покушавају се редом испразнити контејнери почев од првог, по пакетима идући од свог последњег пакета према првом. Приликом пребацивања сваког појединачног пакета у контејнер, при бирању где се пребацује, креће се од последњег у сортираном низу контејнера. Уколико пакет не може да се пребаци у жељени контејнер покушава да се замени са најмањим пакетом тог контејнера мањим од себе са којим је замена могућа. Делимична пребацивања контејнера се такође прихватају, као и у фази размрдавања. Овим поступком се покушава пребацивање и пражњење редом свих контејнера, а сам поступак се може поновити више пута, број понављања је *niter*, један од параметара методе. На основу рачунања функције циља побољшање настаје када се бар један контејнер у потпуности испразни. На крају локалне претраге се као вид побољшања сваком контејнеру покушава смањити тип уколико скуп његових пакета испуњава ограничења масе и запремине. На тај начин се добија мања укупна цена када је то могуће.

По обављеној локалној претрази упоређује се њено најбоље локално решење са укупно најбољим решењем, па се боље од њих задржава за даљи рад (фаза се на енглеском зове Move or Not). Уколико је резултат локалне претраге решење са истом вредношћу функције циља као до тада најбоље решење, у односу на класични облик методе VNS уведена још једна модификација: узима се оно решење које има већи број сасвим пуних контејнера. Сасвим пуни контејнери су они код којих је присутним пакетима до краја попуњен капацитет масе и запремине.

Променљива red, којом се регулише ниво размрдавања, полази увек од 1. Уколико се после обављеног размрдавања за тај ред, и потом обављене локалне претраге, добије решење боље од до тада познатог, онда се поново креће од тог решења; red добија вредност 1 и понавља се поступак. Уколико се не добије боље решење, red се повећава на 2, ако се опет не добије боље решење, на 3, и све тако до rmax. Уколико се ни за rmax не добије боље решење, онда се то броји као једна итерација без поправке решења. Променљива red се враћа на 1 и поступак почиње поново коришћењем најбољег познатог решења. Када се после локалне претраге, без обзира колики је био red, добије боље решење, red се постави на 1 и креће се од почетка. Тада се броје размрдавања, али се то не сматра новом итерацијом. Дакле, у терминима VNS, итерација је везана за постизање rmax вредности променљиве red, а да се најбоље решење није поправило. Такође, приликом повећања вредности променљиве red, до тада урађене измене се поништавају и поново се на описани начин мења најбоље познато решење.

Приметимо да ова три потеза, у оквиру фазе размрдавања, омогућавају широк степен слободе код промене типа контејнера и пребацивања пакета из једног у други контејнер. На тај начин се очекује да се може добити велики број могућности, па ће се тиме превазићи недостаци иницијалног решења, или опасност упадања у локални оптимум. Докле год мале поправке, за мали red, дају боље решење, остаје се при малим редовима. И када се нађе боље решење, такође се остаје при малим вредностима за red. Тек приликом неуспеха поправке, red се повећава. Такође, за разлику од методе GRASP, овде се поправља само најбоље познато решење и не креће се у свакој новој итерацији од почетка.

Што се тиче критеријума заустављања, ту се могу користити разни начини, као на пример укупан број итерација, или укупан број узастопних итерација без поправке решења, или укупно време рада. Овде је, ради лакшег поређења са методом GRASP, као критеријум заустављања узето време рада, исто као и тамо. Овде треба имати у виду да се остајање у границама времена прати приликом уласка у фазу размрдавања за задати ред. Значи, ако је размрдавање дуже трајало, а потом следи и локална претрага, може доћи до прекорачења задатог времена. Приликом тестирања, у пракси се показало да су ово била мала прекорачења која се изражавају у деловима секунде. Могућ је и други приступ код провере времена - да се она врши на крају рада. Тада се не би при-

хватала пронађена решења код којих је прекорачено време. Овде је изабран начин да се провера времена врши на почетку, као што је углавном у литератури.

Описана локална претрага не одговара класичном типу локалне претраге, она циљано сужава простар претраге на оне делове где се очекује да пронађе боље решење и можемо је описати као један вид побољшања онога што се добија у фази размрдавања. Тип методе VNS где се после фазе размрдавања не налази локална претрага назива се редуковани VNS, скраћено RVNS. Алгоритам који је овде предложен описује се као RVNS са побољшањем. То побољшање је код њега уместо класичне локалне претраге, и овде је термин локална претрага коришћен управо за њега као модификовани вид локалне претраге који представља побољшање након фазе размрдавања.

Детаљнији шематски приказ описаног алгоритма дат је као Алгоритам 8. Ознаке које су коришћене имају следеће значење:

- $random(a, b)$ - случајно генерисани цео број из интервала $[a, b]$
- $nBin(S)$ - број контејнера (бинова) у матрици решења S
- $nItem(S, j)$ - број пакета у контејнеру j у матрици решења S
- $load(S, j) = \sum_{i=1}^{n} p_{ija_j} \cdot (m_i, V_i)$ - укупна попуњенст контејнера j у матрици решења S, где је a_j тип контејнера j, док p_{ija_j} је 1 уколико се пакет i смешта у контејнер j иначе је 0
- $f(S, a)$ - вредност функције циља, S је матрица решења, a је низ који памти типове контејнера
- $nFBin(S, a)$ - број пуних контејнера тј. таквих контејнера j код којих је $load(S, j) = (Lm_{a_j}, LV_{a_j})$
- $Swap(S, j_1, i_1, j_2, i_2)$ се односи на замену пакета i_1 из контејнера j_1 и пакета i_2 из контејнера j_2 у ширем смислу, прецизније:

  Ако $j_1, j_2 > 0$**,** онда

  ако $i_1 > 0$ **и** $i_2 > 0$**,** онда замени (ако је могуће) пакет i_1 из контејнера j_1 и пакета i_2 из контејнера j_2;

  ако $i_1 = 0$ **и** $i_2 > 0$**,** онда пребаци (ако је могуће) пакет i_2 из контејнера j_2 у контејнер j_1;

Алгоритам 8 Предложени RVNS алгоритам

procedure RVNS($Problem\ Data, r_{max}, t_{max}$)
  Генерисање иницијалног решења (S, a);
  repeat
    $r \leftarrow 1$;
    while $r \leq r_{max}$ **do**
      $(S', a') \leftarrow (S, a)$; *//фаза размрдавања*
      for $(k \leftarrow 1; k \leq r; k++)$ **do**
        //случајна промена типа случајно изабраног контејнера
        $(j, t) \leftarrow (random(1, nBin(S')), random(1, nt))$;
        if $(load(S', j) \leq capacity(t))$ **then** $a'(j) \leftarrow t$;
        //замена два случ. бирана пакета из два случ. бирана конт.
        $(j_1, j_2) \leftarrow (random(0, nBin(S')), random(0, nBin(S')))$;
        $(i_1, i_2) \leftarrow (random(0, nItem(S', j_1)), random(0, nItem(S', j_2)))$;
        $S' \leftarrow Swap(S', j_1, i_1, j_2, i_2)$;
      for $(k \leftarrow 1; k \leq r; k++)$ **do**
        //покушај пражњења случајно бираног контејнера
        $j \leftarrow random(1, nBin(S'))$;
        $S' \leftarrow EmptyAMAP(S', j)$;
      if (S', a) is not changed **then**
        $r \leftarrow r + 1$;
        continue;
      $(S'', a'') \leftarrow sorted(S', a')$; *//Побољшање*
      for $(iter \leftarrow 1; iter \leq niter; iter++)$ **do**
        for $(j \leftarrow 1; j \leq nBin(S''); j++)$ **do**
          for $(i \leftarrow nItem(S'', j); i \geq 1; i--)$ **do**
            for $(k \leftarrow nBin(S''); k \geq 1; k--)$ **do**
              if $(TransferOrSwap(S'', j, i, k))$ **then**
                break;
          if $(EmptyBin(S'', j))$ **then**
            $(S'', a'') \leftarrow sorted(S'', a'')$;
      $(S'', a'') \leftarrow ImproveByType(S'', a'')$;
      if $f(S'', a'') < f(S, a)$ **then** *//Move or Not*
        $(S, a) \leftarrow (S'', a'')$;
        $r \leftarrow 1$;
      else
        if $(f(S'', a'') = f(S, a)\ and\ nFBin(S'') \geq nFBin(S))$ **then**
          $(S, a) \leftarrow sorted(S'', a'')$;
        $r \leftarrow r + 1$;
  until $SessionTime \geq t_{max}$

ако $i_2 = 0$ **и** $i_1 > 0$**,** онда пребаци (ако је могуће) пакет i_1 из контејнера j_1 у контејнер j_2;

ако $i_1 = i_2 = 0$**,** онда не ради ништа;

ако $j_1 = 0$**,** $j_2 > 0$**, и** $i_2 > 0$**,** онда повећај $nBin(S)$, онда пребаци (ако је могуће) пакет i_2 из контејнера j_2 у нови празан контејнер;

ако $j_2 = 0$**,** $j_1 > 0$**, и** $i_1 > 0$**,** онда повећај $nBin(S)$, и онда пребаци (ако је могуће) пакет i_1 из контејнера j_1 у нови празан контејнер;

ако $j_1 = j_2 = 0$**,** онда не ради ништа.

- $EmptyAMAP(S, j)$ - пребацивање што је више могуће (енгл. as much as possible - AMAP) пакета из j-тог контејнера, $j \in [nBin(S)]$, у остале контејнере $i \in [nBin(S)] \setminus \{j\}$ кренувши од последњег. После примене $EmptyAMAP(S, j)$ контејнер j може да буде испражњен.

- $sorted(S, a)$ - сортирање пакета у сваком контејнеру растуће лексикографски по запремини и маси, као и сортирање скупа контејнера растуће према укупној запремини коју носе

- $TransferOrSwap(S, j, i, k)$ се односи на процедуру која:

  – ако пакет i из контејнера j може да стане у контејнер k, онда се пребацује из контејнера j у контејнер k;

  – у супротном, за сваки пакет l, $l \in [nItem(S, k)]$ контејнера k, покушава се замена пакета i из контејнера j и пакета l из контејнера k док не дође до успешне замене; у том случају она враћа $True$; у супротном враћа $False$. Прецизније, у циљу што већег пражњења разматраног контејнера j допуштена је замена пакета i само са пакетом l који је мање запремине од њега.

Ова локална претрага се разликује код оне која је описана код методе GRASP и описана је као Алгоритам 5 по неколико елемената:

- Цео покушај пражњења који чини локалну претрагу код GRASP овде се понавља више пута ($niter$).

- У случају да пребацивање пакета није могуће покушава се замена пакета, што код GRASP није случај.

- Ако се контејнер само делимично испразнио, овде се то задржава и наставља, док се код GRASP делимична пражњења поништавају.

- Код GRASP се на почетку локалне претраге са вероватноћом од 50% покушава смањити тип контејнера где год је то могуће, док се овде то не ради.

- Начини сортирања нису исти, код VNS је то лексикографски по запремини па маси, док је код GRASP само по маси.

4.6.3 Имплементиране варијанте и тестирање њихових параметара

У фази размрдавања су поменута три начина модификације решења која се раде један за другим. Други начин је подразумевао да се случајно изаберу два различита контејнера и два њихова пакета који замењују места уколико је то могуће. Уочено је да се овај начин може унапредити тако што се дозвољава да контејнер понуди "празан пакет" за размену. У том случају ће тај контејнер у размени добити пакет, али неће дати свој пакет другом контејнеру, па је то уствари пребацивање пакета из једног у други контејнер. Увођење те могућности омогућило је да се, приликом бирања, уведе нови контејнер у састав решења, при чему он као празан може да понуди само празан пакет за размену, односно да добије пакет из неког другог контејнера. Избор новог празног контејнера се дешава са подједнаком вероватноћом као избор било ког другог контејнера који већ припада решењу. Приликом тестирања се показало да овај корак са празним пакетом значајно поправља квалитет фазе размрдавања и уопште методе VNS. Како се пакети у контејнеру бирају са подједнаком вероватноћом, поставило се питање колику вероватноћу доделити избору празног пакета. Један приступ је подразумевао да је вероватноћа избора празног пакета у контејнеру иста као и свих осталих његових пакета, односно да је релативна и зависи од постојећег броја пакета у изабраном контејнеру. Нпр. ако их има br, онда је $1/(br+1)$ вероватноћа да се изабере "празан" пакет. Ова варијанта методе VNS у наставку се реферише као VNS-1. Други приступ подразумева да се код празног контејнера увек бира празан пакет, али код контејнера који већ садржи пакете за избор празног пакета не користи се релативна вероватноћа него нека фиксно одређена вероватноћа. Ова варијанта методе VNS у наставку се реферише као VNS-2. И један и други приступ имају своју логику и оправдање. Релативна вероватноћа избора празног пакета је на први поглед логична, али она може значајно да варира и да то утиче на квалитет добијеног решења. Код фиксне вероватноће

избора празног пакета, остали "реални" пакети равноправно деле преосталу вредност вероватноће, нпр. ако је вероватноћа избора празног пакета фиксирана на 20%, остали пакети равномерно деле преосталих 80% вероватноће. Метода VNS-2 као параметар има p, фиксну вероватноћу избора празног пакета, док параметар rmax постоји код обе методе. Обе варијанте карактерише и параметар *niter* из локалне претраге, колико пута се извршава циклус којим се покушава пражњење контејнера.

Имплементација је урађена у програмском језику C. Тестирање параматара спроведено је подскупу од 10 великих инстанци, узета је по једна инстанца величина 50, 70, 100, 120, 150, 200, 350, 500, 750 и 1000 пакета. Свако извршавање програма је трајало 20 секунди и потом поновљено 10 пута за изабрану инстанцу и вредности параметара методе. Избор параметра *rmax* представља посебан изазов. Могу се изабрати константне вредности, али то онда носи опасност да за велике инстанце, у фази размрдавања, неће моћи довољно добро да се измени решење, посебно ако је *rmax* мало. Зато има смисла да *rmax* буде довољно велико и сразмерно броју пакета n. За вредности параметра *rmax* узимане су могућности из скупа $rmax = \lfloor 0,05kn \rfloor$, $1 \leq k \leq 10$, где је n број пакета текуће инстанце. За вредности параметра *niter* узимани су природни бројеви од 1 до 10, тј. $niter \in \{1, \ldots, 10\}$.

Код методе VNS-1 оба параметра су детаљно тестирана. У прелиминарним тестирањима, која овде не наводимо детаљно, вредност $rmax = \lfloor 0,05kn \rfloor$ за $k = 5$ се показала нешто боља од осталих, зато је приликом тестирања параметра *niter* вредност *rmax* фиксирана управо на $rmax = \lfloor 0,25n \rfloor$. Резултати тестирања параметра *niter* дати су у табели 24. Подебљано су приказане вредности које су најбоље (најмање) за инстанцу која се приказује у тој колони табеле. Ту се може закључити да је вредност $niter = 6$ показала најбоље резултате јер је код свих 10 тестираних инстанци постигла најбоље решење. Након тога, за фиксирану вредност параметра $niter = 6$ спроведено је тестирање параметра *rmax* за свих 10 разматраних вредности. Резултати овог тестирања приказани су у табели 25. Ту се може закључити да су за вредност $rmax = \lfloor 0,25n \rfloor$ добијена најбоља решења за сваку од 10 разматраних инстанци. На тај начин је за методу VNS-1 у наставку узето да су вредности њених параметара $niter = 6$ и $rmax = \lfloor 0,25n \rfloor$.

Метода VNS-2 се од VNS-1 разликује по вероватноћи избора "празног

n: *niter*	50	70	100	120	150
1	**31517**	47094	64291	**65906**	86423
2	**31517**	47094	64291	**65906**	**85586**
3	**31517**	47094	64291	**65906**	**85586**
4	**31517**	47094	**63454**	**65906**	**85586**
5	**31517**	47094	**63454**	**65906**	**85586**
6	**31517**	**46257**	**63454**	**65906**	**85586**
7	**31517**	47094	**63454**	**65906**	**85586**
8	**31517**	47094	**63454**	**65906**	**85586**
9	**31517**	47094	**63454**	**65906**	**85586**
10	**31517**	47094	**63454**	**65906**	**85586**
n: *niter*	200	350	500	750	1000
1	120226	209653	296247	444569	596963
2	119402	207966	294560	443719	595395
3	**119389**	208803	294560	442151	594545
4	**119389**	**207248**	293723	**440477**	593721
5	**119389**	**207248**	293723	**440477**	593708
6	**119389**	**207248**	**292886**	**440477**	**592871**
7	**119389**	**207248**	**292886**	**440477**	592884
8	**119389**	**207248**	293723	**440477**	**592871**
9	**119389**	**207248**	293723	**440477**	**592871**
10	**119389**	**207248**	**292886**	**440477**	**592871**

Табела 24: Тестирање параметра *niter* за методу VNS-1 за 20s

пакета" p (тј. вероватноћи пребацивања уместо замене пакета) у фази размрдавања. Зато метода VNS-2 осим параметара *niter* и *rmax* има и параметар p. Пошто је у највећем делу метода VNS-2 иста са VNS-1 онда су узете исте фиксиране вредности параметара *niter* и *rmax*, а додатно је тестиран параметар p. Вредности параметра p су узимане као константе из скупа $p = 0,05k$, $1 \leq k \leq 10$. Тестирање је извршено под истим условима као за *niter* и *rmax*, на истом скупу од 10 великих инстанци, са по 10 извршавања од по 20 секунди рада. Резултати тестирања приказани су у табели 26. На основу тога као најбоље издвајају се вредности $0,05$ и $0,2$. Међу њима је за даље изабрана вредност $0,25$ параметра p. Између осталог вредност параметра $p = 0,2$ коришћена је у раду [62] из 2018. године где су описане сличне методе VNS-1 и VNS-2, које су овде побољшане. Након тестирања параметара, избором $niter = 6$, $rmax = \lfloor 0,25n \rfloor$ и $p = 0,2$ одређена је метода VNS-2.

n: r_{max}	50	70	100	120	150
$\lfloor 0.05n \rfloor$	32354	47931	64291	66743	86423
$\lfloor 0.10n \rfloor$	**31517**	47094	**63454**	**65906**	86423
$\lfloor 0.15n \rfloor$	**31517**	47094	**63454**	**65906**	86423
$\lfloor 0.20n \rfloor$	**31517**	47094	**63454**	**65906**	**85586**
$\lfloor 0.25n \rfloor$	**31517**	**46257**	**63454**	**65906**	**85586**
$\lfloor 0.30n \rfloor$	**31517**	47094	**63454**	**65906**	**85586**
$\lfloor 0.35n \rfloor$	**31517**	47094	**63454**	**65906**	**85586**
$\lfloor 0.40n \rfloor$	**31517**	47107	**63454**	**65906**	**85586**
$\lfloor 0.45n \rfloor$	**31517**	47094	**63454**	**65906**	**85586**
$\lfloor 0.50n \rfloor$	**31517**	47094	**63454**	**65906**	**85586**
n: r_{max}	200	350	500	750	1000
$\lfloor 0.05n \rfloor$	121063	209640	295397	442151	596219
$\lfloor 0.10n \rfloor$	120226	208803	**292886**	441314	593708
$\lfloor 0.15n \rfloor$	120226	208803	293723	441314	593708
$\lfloor 0.20n \rfloor$	**119389**	**207248**	293723	**440477**	593708
$\lfloor 0.25n \rfloor$	**119389**	**207248**	**292886**	**440477**	**592871**
$\lfloor 0.30n \rfloor$	**119389**	**207248**	293723	**440477**	**592871**
$\lfloor 0.35n \rfloor$	**119389**	**207248**	293723	**440477**	**592871**
$\lfloor 0.40n \rfloor$	**119389**	207979	**292886**	**440477**	**592871**
$\lfloor 0.45n \rfloor$	**119389**	208803	293723	**440477**	593708
$\lfloor 0.50n \rfloor$	**119389**	207966	293723	**440477**	**592871**

Табела 25: Тестирање параметра *rmax* за методу VNS-1 за 20s

4.6.4 Добијени резултати

За тестирање су коришћени исти скупови инстанци као и код методе GRASP, односно решавача CPLEX. Задавана су временска ограничења по једном извршењу. Програми су покретани 30 пута и бележена су најбоља решења из тих 30 пролазака. Параметри *agap* и σ рачунати су на исти начин као код методе GRASP. Такође, бележено је и просечно време потребно да се нађе најбоље решење, као и просечан број размрдавања, односно итерација у складу са описаним појмом итерације код VNS. Овде нису бележени просечни бројеви размрдавања и итерација до најбољег решења јер су они, углавном, сразмерни времену потребном за налажење најбољег решења. Ово време се прати као главни параметар рада. Осим тога, избегнуто је оптерећивање табела великим бројем података.

Из свих 6 мањих инстанци, за време рада ограничено на 1 секунду по извршењу, код обе варијанте VNS за време знатно краће од 1 секунде,

n: p	50	70	100	120	150
0.05	**31517**	**47094**	**63454**	**65906**	**85586**
0.10	**31517**	**47094**	**63454**	**65906**	**85586**
0.15	**31517**	**47094**	**63454**	**65906**	**85586**
0.20	**31517**	**47094**	**63454**	**65906**	**85586**
0.25	**31517**	**47094**	**63454**	**65906**	**85586**
0.30	**31517**	**47094**	**63454**	**65906**	**85586**
0.35	**31517**	**47094**	**63454**	**65906**	**85586**
0.40	**31517**	**47094**	**63454**	**65906**	**85586**
0.45	**31517**	**47094**	64291	**65906**	**85586**
0.50	**31517**	**47094**	**63454**	**65906**	**85586**
n: p	200	350	500	750	1000
0.05	**119389**	**207248**	**292886**	**440477**	**592871**
0.10	**119389**	**207248**	293723	**440477**	**592871**
0.15	**119389**	207261	**292886**	**440477**	**592871**
0.20	**119389**	**207248**	**292886**	**440477**	**592871**
0.25	**119389**	207966	**292886**	**440477**	**592871**
0.30	120226	**207248**	**292886**	**440477**	593708
0.35	**119389**	**207248**	293723	**440477**	593708
0.40	**119389**	**207248**	293723	**440477**	**592871**
0.45	**119389**	**207248**	**292886**	**440477**	593708
0.50	**119389**	**207248**	293723	441314	**592871**

Табела 26: Тестирање параметра p за методу VNS-2 за 20s

генерисана су иста оптимална решења која су добили CPLEX и GRASP. Код свих шест инстанци, у свих 30 тестирања, добијено је истоветно решење. Добијени подаци налазе се у табелама 27 и 28. Овде је за разлику од наредних табела забележен и број просечних итерација до најбољег решења.

За називе колона коришћени су скраћене ознаке:

- *најр* - најбоље решење,
- *пвдн* - просечно време до налажења најбољег решења,
- *пубр* - просечан укупан број извршених размрдавања за задато време,
- *пуби* - просечан укупан број извршених итерација за задато време,
- *пидн* - просечан број итерација који је потребан да се добије најбоље решење.

бр. пакета	*најр*	*пвдн*	*пубр*	*пуби*	*пидн*
10	6534	0,0002	119500	65641	9,633
11	7252	0,0	135516	74703	0,600
12	8846	0,0	47283	16448	0,033
13	9722	0,0002	64356	24136	2,067
15	12047	0,0123	36449	12487	128,07
20	13786	0,0037	36213	7432,0	20,133

Табела 27: Добијена решења методом VNS-1 за 1s

бр. пакета	*најр*	*пвдн*	*пубр*	*пуби*	*пидн*
10	6534	0,0002	117852	64454	6,467
11	7252	0,0	135785	74208	0,833
12	8846	0,0	47122	16329	0,067
13	9722	0,0002	63241	23543	0,967
15	12047	0,0130	36652	12540	137,50
20	13786	0,0032	36341	7451,5	17,067

Табела 28: Добијена решења методом VNS-2 за 1s

Како је метода успешно примењена и дала добре резултате за мање инстанце, закључује се да је њена употреба оправдана за веће инстанце које су потом детаљније тестиране.

У табелама 29 и 30 детаљније су приказани резултати рада метода VNS-1 и VNS-2 за 60 секунди по извршењу, на скупу од 10 великих инстанци величина од 50 до 1000 пакета. У њима су за називе колона, које се поклапају са описима података у претходним табелама, коришћени скраћени записи: *пвдн.* - просечно време до најбољег, *пбразм.* - просечан укупан број размрдавања, *пбитер.* - просечан укупан број итерација, *пидн.* - просечан број итерација до најбољег решења.

бр. пак.	најб. реш.	пвдн.	пбразм.	пбитер.	agap	σ
50	31517	8,081	1392688	122530	0,000	0,000
70	46257	23,786	713832	43604	2,055	2,195
100	63454	13,680	1174193	49531	0,704	0,963
120	65906	15,454	1026241	36001	0,000	0,000
150	85586	23,232	773175	21766	0,163	0,399
200	119389	20,284	489504	10086	0,117	0,286
350	206411	35,536	246257	2879,6	0,454	0,522
500	292886	23,770	170235	1374,6	0,143	0,202
750	439640	30,910	100363	533,10	0,178	0,190
1000	592871	32,919	58551	226,83	0,047	0,082

Табела 29: Добијена решења помоћу VNS-1 за 60s

бр. пак.	најб. реш.	пвдн.	пбразм.	пбитер.	agap	σ
50	31517	9,691	1314684	115480	0,001	0,008
70	47094	20,596	710157	43384	0,302	0,728
100	63454	19,682	1137134	47946	0,484	0,799
120	65906	12,812	1029870	36125	0,000	0,000
150	85586	24,034	764693	21517	0,196	0,437
200	119389	27,750	473687	9756,8	0,000	0,000
350	206411	30,428	255144	2984,3	0,467	0,526
500	292886	29,596	169431	1368,1	0,105	0,173
750	439640	28,547	97404	517,23	0,222	0,243
1000	592871	32,771	57859	224,03	0,033	0,068

Табела 30: Добијена решења помоћу VNS-2 за 60s

бр. пак.	CPLEX 1h	CPLEX LB	VNS-1	VNS-2	U-GRASP
50	31530	30398,95	31517	31517	32367
70	48688	45349,16	46257	47094	47996
100	65898	62834,24	63454	63454	66774
120	68376	65301,77	65906	65906	70076
150	88056	84667,92	85586	85586	91482
200	124394	118014,05	119389	119389	128631
350	219321	206110,49	206411	206411	224475
500	310801	291891,06	292886	292886	317149
750	4910250	439092,26	439640	439640	477587
1000	6547000	591583,60	592871	592871	641267

Табела 31: Добијена решења помоћу CPLEX, VNS и GRASP за 60/120 s

У табели 31 дат је упоредни приказ добијених решења на 10 великих инстанци, од 50 до 1000 пакета, за VNS-1 и VNS-2 за 60 секунди рада по једном извршењу. Приказани су заједно са решењима до којих су дошли CPLEX за 1 сат и U-GRASP за 120 секунди по извршењу. Наведена решења CPLEX нису гарантовано оптимална, у табели су наведене и вредности доњих граница до којих је CPLEX дошао решавајући их. Ако се међусобно упореде VNS-1 и VNS-2 може се видети да су на 9 инстанци добиле иста решења, док је на 1 инстанци VNS-1 добила боље решење. У поређењу са GRASP, обе методе VNS-1 и VNS-2 биле су боље од ње на свим инстанцама. Поредећи их са CPLEX закључујемо да су обе методе VNS-1 и VNS-2 биле боље на свих 10 инстанци. Из табеле се може закључити да се помоћу GRASP и VNS могу решавати велике инстанце које CPLEX не може добро да реши или не може да их решава. На основу добијених резултата се закључује да су обе варијанте VNS доброг квалитета и оправдане за даље тестирање, међу њима је VNS-1 нешто

боља. У наставку је за тестирања VNS коришћена само VNS-1, па ће се навођење само VNS односити на њу.

Због сличне структуре метода VNS и GRASP погодно је упоредити шта се добија разменом већих компоненти ових метода. У табели 32 дат је приказ резултата за стандардну варијанту VNS, потом варијанте где је код VNS уместо сопствене локална претрага из GRASP (VNS-LG), затим где је уместо иницијалног решења примењена процедура GRC (VNS-IG) или где су оба поменута дела замењена (VNS-ILG). Такође у истој табели је осим резултата GRASP приказан и резултат који се добија када уместо њене дође локална претрага из VNS (GRASP-LV). Иницијално решење код VNS није од великог значаја у односу на локалну претрагу јер се извршава само једном на почетку. Локална претрага из GRASP није поправила методу VNS. Међутим и код методе GRASP када је постављена локална претрага из VNS добила су се лошија решења него пре тога. Ту можемо да закључимо да различита природа ових метода издаваја различите локалне претраге као боље, тако да је постојање две локалне претраге за њих оправдано. Локална претрага VNS дуже траје и због итеративне природе методе решење се поправља поновном претрагом. За разлику од тога методи GRASP погодује локална претрага која је брза, која за кратко време даје најбоље што може, а како се претрага не наставља одатле онда јој краће локалне претраге дају могућност да метода спроведе више својих итерација и тако кроз већи број покушаја нађе боље решење.

бр. пак.	VNS	VNS-LG	VNS-IG	VNS-ILG	GRASP	GRASP-LV
50	**31517**	32354	**31517**	**31517**	32367	32235
70	**46257**	48768	47094	48768	47996	48688
100	**63454**	65965	**63454**	65965	66774	66748
120	**65906**	66743	**65906**	66743	70076	74482
150	**85586**	88097	**85586**	88097	91482	94828
200	**119389**	122737	**119389**	122737	128631	134854
350	**206411**	213851	**206411**	213838	224475	240503
500	**292886**	304524	**292886**	304537	317149	342471
750	**439640**	459711	440477	463751	477587	509749
1000	**592871**	622093	**592871**	630500	641267	690551

Табела 32: Добијена решења помоћу различитих локалних претрага и почетних решења VNS за 60s и GRASP за 120s

Будући да су решења добијена методом VNS високог квалитета и осетно боља од оних до којих је дошао CPLEX, а пошто CPLEX губи доста

Инстанца	**CPLEX** 1800s		**VNS** 1s		**VNS** 60s		**U-GRASP** 120s	
n_i	*реш.*	*LB*	*реш.*	*gap*(%)	*реш.*	*gap*(%)	*реш.*	*gap*(%)
50_1	31530	30398,95	**31517**	1,70	**31517**	0,00	32367	0,05
50_2	31491	29368,89	**30615**	0,30	**30615**	0,00	31359	0,45
50_3	32577	30721,37	**31009**	5.03	**31009**	0,99	32577	0,02
50_4	27624	26393,31	**26761**	2,50	**26761**	0,00	27611	0,14
50_5	**28500**	27382,40	**28500**	0,00	**28500**	0,00	28513	0,03
70_1	47996	45349,16	47107	1,90	**46257**	2,06	47996	1,51
70_2	44624	42550,00	**43761**	2,36	**43761**	0,57	45487	0,05
70_3	43564	41353,78	**42675**	1,57	**42675**	0,00	44388	0,04
70_4	43577	40751,86	**41957**	1,49	**41957**	0,00	43696	0,82
70_5	43932	41871,11	**43030**	1,05	**43030**	0,00	44637	0,25
100_1	65048	62834,24	64291	0,83	**63454**	0,70	66774	1,09
100_2	61453	58350,92	**58996**	1,61	**58996**	0,00	62303	1,10
100_3	58812	55537,24	**56329**	1,75	**56329**	0,00	59610	0,23
100_4	64654	61209,88	62184	2,16	**62171**	0,85	65491	0,81
100_5	64444	60751,35	62666	0,97	**61961**	0,46	66077	0,90
120_1	69252	65301,77	**65906**	1,66	**65906**	0,00	70076	0,39
120_2	79355	74939,40	76859	1,27	**76009**	1,14	81055	0,81
120_3	76206	72707,13	74560	0,94	**73723**	0,76	77201	0,86
120_4	72269	69002,17	69799	2,15	**69773**	0,64	73982	1,45
120_5	73526	70048,06	**71043**	1,66	**71043**	0,12	75918	0,28
150_1	88958	84667,92	86423	0,81	**85586**	0,16	91482	0,87
150_2	94787	91176,05	**92291**	1,00	**92291**	0,00	97231	1,37
150_3	88437	84982,07	86778	0,26	**85941**	0,00	90137	1,08
150_4	93911	89060,07	90684	1,24	**89847**	0,56	95624	0,83
150_5	89650	85053,45	**86436**	0,74	**86436**	0,00	91324	0,81
200_1	124394	118014,05	120239	0,59	**119389**	0,12	128631	0,69
200_2	123137	116939,41	118995	0,50	**118158**	0,05	127361	0,36
200_3	126496	121239,39	123230	1,20	**122393**	0,30	132366	0,87
200_4	119428	111622,52	113863	0,25	**113026**	0,00	119601	0,80
200_5	125436	119045,81	121281	0,47	**120431**	0,14	129580	0,52
350_1	225215	206110,49	207979	1,11	**206411**	0,45	224475	0,41
350_2	227514	210898,41	213940	0,69	**212253**	0,12	228762	0,41
350_3	217797	201371,73	202168	1,41	**202155**	0,01	218796	0,38
350_4	219702	206178,19	208308	0,62	**207471**	0,00	223236	0,64
350_5	219023	207275,65	209197	1,22	**208347**	0,01	224257	1,00
500_1	318066	291891,06	294560	0,56	**292886**	0,14	317149	0,52
500_2	339309	294525,03	296364	0,67	**295527**	0,01	320627	0,44
500_3	361838	284648,85	286616	0,81	**284929**	0,25	307855	0,47
500_4	382560	308367,53	312765	0,71	**310215**	0,26	335012	0,60
500_5	319323	290365,44	293150	0,53	**291476**	0,08	314376	0,46
750_1	546742	439092,26	443825	0,42	**439640**	0,18	477587	0,39
750_2	537178	442327,90	445842	0,52	**443318**	0,18	479827	0,32
750_3	554712	443803,57	447467	0,47	**445062**	0,05	481910	0,49
750_4	548336	442099,54	447184	0,60	**442986**	0,13	481446	0,40
750_5	551524	448636,78	452394	0,67	**449857**	0,07	488107	0,40
1000_1	739616	0	597906	0,46	**592871**	0,05	641267	0,49
1000_2	733240	0	589508	0,55	**582799**	0,14	631597	0,45
1000_3	750774	0	606806	0,51	**598529**	0,42	651655	0,43
1000_4	728458	0	601589	0,35	**594999**	0,18	645445	0,44
1000_5	744398	0	606640	1,23	**599094**	0,07	649685	0,33

Табела 33: Поређење резултата на великим инстанцама

времена долазећи до својих решења, у наставку је примењен још један приступ. Наиме, иницијална решења до којих долази VNS на почетку свог рада дају се као полазна - иницијална решења CPLEX-у. Тиме се решавач значајно помера од почетка и већи су изгледи да добије боље решење. У табели 33 дати су резултати тестирања CPLEX за време од 1800 секунди са иницијалним решењем које је као у методи VNS, затим решења добијена методом VNS за 1 и 60 секунди по извршењу, као и решења добијена методом U-GRASP за 120 секунди рада по извршењу. Примећује се да повећање времена рада методе VNS доприноси налажењу бољих решења, што је нарочито приметно код великих инстанци. Код свих разматраних инстанци методом VNS за 60 секунди добило се најбоље решење. Метода U-GRASP обезбеђује стабилна решења, али мањег квалитета.

У табели 33 може се уочити да је метода VNS за 60 секунди рада достигла вредности врло близу гарантованих доњих граница до којих је дошао CPLEX. То је био мотив да се покуша са налажењег гарантовано оптималних решења за ове инстанце помоћу CPLEX. Тако је повећано време рада решавача на 1 сат користећи паралелизацију на 8 доступних језгара, а као иницијална решења су узета најбоља до којих је дошао VNS. У овом поступку CPLEX није дошао до бољих решења осим што је код инстанце 50_5 доказао да је за њу добијено решење методом VNS оптимално.

Након тога учињен је још један покушај да се испита оптималност добијених решења. Како су цене превоза контејнера цели бројеви (1594, 2470, 2483), укупна цена превоза скупа контејнера увек је број облика $C = 1594a + 2470b + 2483c$, где је a, b, c број контејнера сваког од три типа. Ако са LB означимо доњу границу из CPLEX, а са $RVNS$ решење добијено методом VNS, оптимална цена задовољава услов $LB \leq C \leq RVNS$ уз који се додаје и услов потребне носивости да тројка броја контејнера сваког типа (a, b, c) по укупној маси и запремини има капацитет за све пакете. Овај други услов се може записати формулама овако: $\sum m_i \leq 25.8a + 24.5b + 24.5c$ и $\sum V_i \leq 30a + 60b + 70c$. Он је само потребан, али не и довољан јер и када постоји укупни капацитет не значи да ће пакети моћи да се распореде јер нема поделе пакета, тако да овај услов може да дода неке немогуће тројке у кандидате за оптимално решење. Ако би постојала само једна тројка (a, b, c) за коју важе сви поменути услови онда је она једини могући избор за оптимално решење, па би

тиме била доказана оптималност добијеног решења. Применом овог поступка доказана је оптималност добијених решења VNS за инстанце 50_3, 50_4, 100_1, 100_2, 120_1 и 350_1, што заједно са 50_5 даје укупно 7 гарантовано оптимално решених инстанци.

Додатним повећањем времена рада CPLEX са 1 на 5 сати коришћењем паралелизације на 8 језгара и полазећи од најбољих решења из VNS није добијена поправка ни једног решења, као ни доњих граница за CPLEX изузев доњих граница за инстанце са 1000 пакета. Добијене доње границе за њих су: $591583,60$ за 1000_1; $581894,37$ за 1000_2, $597452,82$ за 1000_3; $594256,58$ за 1000_4 и $597665,20$ за 1000_5. Може се уочити да су ове доње границе врло близу оних решења до којих је дошао VNS.

Како CPLEX није успео да докаже оптималност добијених решења онда је значајно повећано време рада методе VNS. Повећањем са 60 на 360 секунди по извршењу добила су се боља решења код 4 инстанце: 100_5 (61124), 500_4 (309378), 750_3 (444225) и 750_5 (449020). Додатним повећањем са 360 на 1440 секунди по извршењу добило се побољшање решења код још једне инстанце: 1000_3 (597692). Потом је за свако од 5 нових добијених решења као иницијалним покренут CPLEX за 5 сати рада (паралелно са 8 језгара). Међутим ни ту се нису добила боља решења нити је доказана оптималност неког од њих, само је једна доња граница повећана, код инстанце 1000_3 на $597452,82$. За нова решења ни метода са уређеним тројкама (a, b, c) није доказала оптималност, јер је у задатим интервалима било више кандидата за оптимално решење.

Добијени резултати ове методе објављени су крајем 2021. године у раду [63].

Описане методе VNS и GRASP су развијане и тестиране користећи цену као изабрану функцију циља, пре свега да би се лакше разумео смисао добијених бројева. Потпуно исти механизам може се употребити како би се пронашао распоред пакета по контејнерима који уместо цене оптимизује загађење или неку њихову комбинацију.

Укупно, може се сматрати да су развијене варијанте методе VNS врло квалитетно успевале да одреде решења, да су на свим инстанцама на тај начин добијена иста или боља решења од оних које је добио CPLEX. Осим тога, њихова предност је што и за велике инстанце за које CPLEX не може да одреди решење, оне могу да нађу одговарајуће приближно решење.

4.7 Решавање проблема паковања помоћу грамзивих хеуристика

Да би се испитао квалитет предложених метода, њихови резултати се пореде са оним што је урађено у раду [2]. Ту су проблеми паковања решавани као проблеми одлучивања тј. за дати скуп пакета и контејнера формира се одговор да ли се они могу спаковати или не, применом алгоритма који је код сваке од метода у некој мери различит, а заједничко им је да имају грамзиви приступ и не користе локалну претрагу нити поновно генерисање решења кроз итерације. Зато све ове методе брзо генеришу решења, али немају задатак да оптимизују број употребљених контејнера, јер је одговор само "да" или "не". Уколико је одговор да, онда су дати пакети спаковани у контејнере по том алгоритму, а ако је одговор не, онда значи да не могу да се спакују по том алгоритму, што не значи да се не могу спаковати на неки други начин.

У том раду су представљене укупно 34 хеуристике за решавање хетерогеног векторског паковања. Оне су систематично задате на следећи начин: Прва подела је била по глобалном приступу грамзивог паковања, где су наведене 4 приступа, по својим енглеским називима то су: item centric, bin centric, bin balancing и single bin balancing. Код пакет центричног приступа (item centric) све док постоји пакет који није спакован прерачунава се слободан простор у контејнерима и пакује највећи пакет у расположиви контејнер са најмање преосталог места и поступак се понавља све док се сви пакети не спакују и тада је одговор "да". Уколико се дође до пакета који не може да се спакује нигде, онда је одговор "не" и прекида се даљи рад програма.

Код контејнер центричног приступа (bin centric) све док постоје контејнери у којима има места, прерачунава се слободан простор у њима, изабере се контејнер са најмање слободног места, а потом се у њега пакују пакети тако што се крене од највећег који може да стане, па се онда прерачунава слободан простор и све тако док бар неки пакет може да се још смести у тај контејнер. Када се поступак паковања тог контејнера заврши, он се брише из списка контејнера који су расположиви (у којима има још места) и понавља се поступак за наредни контејнер. На крају када се тако исцрпе сви контејнери, ако је остао неки пакет који није спакован одговор је "не", у супротном одговор је "да".

Претходна два приступа практично имплицитно теже да смање број употребљених контејнера. Насупрот њима су контејнер балансирајуће хеуристике које имају тежњу да пакете равномерно распоређују у доступне контејнере. Ту се контејнери сортирају у једну листу, као и пакети, па се највећи неспакован пакет покушава сместити у контејнер који је први на реду, па се прелази на следећи пакет и следећи контејнер. Разлика између приступа bin balancing и single bin balancing је у томе што се код првог од њих у случају неуспеха паковања у контејнер он поново разматра за следећи пакет и све тако док се не спакује неки пакет у њега. Код single bin balancing приступа у случају неуспеха, за наредни пакет иде се одмах на следећи контејнер. Овде се контејнери разматрају као да су распоређени у облику кружне листе или у облику низа код којег се управо обрађени контејнер ставља на крај.

Приликом рачунања користе се вредности $C(j)$, преостали капацитети контејнера по ресурсу (димензији) j, и $R(j)$ потребе неспакованих пакета по ресурсу j. Приликом распоређивања коришћено је укупно 9 мера: ниједна (статичка, пакетима и контејнерима се приступа оним редом како су задати на улазу), затим 4 мере које могу бити статичке и динамичке: случајно мешање (shuffle), $1/C$, $1/R$ и R/C. Када се помножи ових 9 мера са 4 приступа добија се 36 метода, међутим овде статичке мере код пакет центричних и контејнер центричних приступа дају потпуно исте резултате, а како таквих мера има 5, то је укупно добијена 31 метода. Осим њих коришћене су и три методе засноване на скаларном производу где се за паковање пакета у контејнер бира онај пар који даје највећу вредност скаларног производа преосталих капацитета контејнера и потреба пакета. Ове три методе разликују се по коефицијентима (скаларима) којима се множе сабирци скаларног производа са циљем да се на неки начин нормирају. Тиме су укупно добијене 34 методе.

Како методе које раде са балансирањем контејнера теже да у сваки доступан контејнер поставе по пакет оне нису од користи приликом рачунања оптималног броја употребљених контејнера. Таквих метода овде је било 18. Осталих 16 метода имплицитно теже да утроше што мањи број контејнера, па су оне овде имплеметиране ради поређења. Осим што су методе у раду [2] примарно предвиђене да утврде да ли се може спаковати дати скуп пакета у дати скуп контејнера, па се не баве директно оптимизацијом броја контејнера, оне у свом изворном облику не користе ни цене за поједине типове контејнера како би се бавиле укупном

мет./инст.	50	70	100	120	150	200	350	500	750	1000
itcen st. n	27	43	57	48	68	92	157	223	339	452
itcen st. shuf.	19	28	36	40	50	69	121	171	249	338
itcen st. 1/C	25	42	55	48	65	89	154	217	333	443
itcen st. 1/R	25	42	55	47	65	89	153	216	331	441
itcen st. R/C	25	42	55	47	65	88	153	216	331	440
itcen din. shuf.	42	65	90	105	129	176	299	438	658	878
itcen din. 1/C	25	42	55	48	65	89	154	217	333	442
itcen din. 1/R	25	42	55	47	65	88	152	221	330	439
itcen din. R/C	25	42	55	47	64	88	153	216	330	441
bincen din. shuf.	20	27	36	42	52	73	130	178	266	355
bincen din. 1/C	25	42	55	48	65	89	154	217	333	443
bincen din. 1/R	25	42	55	47	65	89	155	220	333	447
bincen din. R/C	25	42	54	47	65	88	151	216	329	439
dotprod 1	50	70	100	120	150	200	350	500	750	1000
dotprod 2	27	42	73	91	93	128	227	316	476	669
dotprod 3	25	42	55	47	66	89	155	221	337	448
VNS-1	17	24	32	38	47	66	116	162	239	324
U-GRASP	17	24	33	39	47	67	118	164	243	330

Табела 34: Добијена решења помоћу 16 метода из рада [2], VNS-1 и U-GRASP, на инстанцама са индексом 1

ценом. Поређење је урађено на 50 великих инстанци величина 50, 70, 100, 120, 150, 200, 350, 500, 750 и 1000 пакета. Пошто 16 одабраних метода из рада [2] не могу добро да савладају различите цене контејнера, да бисмо избројали колико је утрошено контејнера постављено је да им је свима једнака цена 1. Са њима су упоређене варијанта методе VNS-1 и U-GRASP прилагођене тако да је цена сваког типа контејнера једнака 1. Резултати се могу видети у табелама 34-38. Врсте табеле одговарају примењеним методама, а колоне одговарају величинама инстанци, изражених у броју пакета. Вредности у табели представљају број утрошених контејнера. Како креирана библиотека има по 5 инстанци сваке величине, резултати су представљени кроз ових 5 табела, према индексу разматраних инстанци. Ту су приказани и резултати рада метода VNS-1 и U-GRASP код којих је као време рада изабрано 2 секунде по извршењу за 8 мањих инстанци, и 5 секунди по извршењу за инстанце од 750 и 1000 пакета. Обе методе су тестиране по 30 пута и приказано је најбоље добијено решење. Овде се може видети да методе VNS-1 и U-GRASP чак и за овако мало време дају добра решења, боља од предложених 16 метода из рада [2]. Осим тога методе из тог рада су коначно

мет./инст.	50	70	100	120	150	200	350	500	750	1000
itcen st. n	24	39	47	65	76	87	167	229	333	451
itcen st. shuf.	18	24	35	45	53	72	125	169	258	328
itcen st. 1/C	23	36	46	61	73	84	159	220	324	440
itcen st. 1/R	23	36	46	60	73	84	159	220	323	437
itcen st. R/C	23	36	46	60	72	84	158	220	323	438
itcen din. shuf.	42	63	90	104	128	174	297	431	657	872
itcen din. 1/C	23	36	46	61	73	84	159	221	324	440
itcen din. 1/R	23	36	46	60	72	84	158	219	322	437
itcen din. R/C	23	36	46	60	72	84	158	220	323	437
bincen din. shuf.	19	26	36	45	55	75	129	177	272	339
bincen din. 1/C	23	36	46	61	73	84	159	220	324	440
bincen din. 1/R	24	36	47	62	74	84	161	222	324	440
bincen din. R/C	22	36	45	59	72	84	157	218	322	436
dotprod 1	50	70	100	120	150	200	350	500	750	1000
dotprod 2	40	45	69	74	98	114	230	319	486	657
dotprod 3	23	36	46	61	73	84	161	224	326	447
VNS-1	17	23	32	41	49	68	117	162	248	316
U-GRASP	17	23	33	41	50	69	119	165	252	321

Табела 35: Добијена решења помоћу 16 метода из рада [2], VNS-1 и U-GRASP, на инстанцама са индексом 2

завршене и не може се повећањем времена утицати на њихов квалитет, док код VNS-1 и U-GRASP се време рада може повећавати и добити још боља решења, нарочито код већих инстанци.

Овде је занимљиво запазити да бројање контејнера код VNS-1 и U-GRASP није приступ где оне показују своју праву снагу јер постоје кораци алгоритма у којима се, ако је могуће, смањује тип контејнера, јер су методе изворно прављене за хетерогени случај када је цена превоза мањег контејнера такође мања. Уколико су цене исте, као када се броје контејнери, онда то смањивање типа може да доведе да повећања потребног броја контејнера и да се спорије проналазе боља решења. Бројање контејнера не иде у прилог ни методама из рада [2] јер оне углавном теже да попуњавају контејнере са најмање слободног места, па ће зато највише да користе мале контејнере, зато ће им бити потребан већи број контејнера. Како код практично свих метода приступ рада највише тежи ка избору малих контејнера, онда је имало смисла упоредити не само добијени број контејнера него и уопште добијену цену потребну да се скуп пакета транспортује према ценама које су коришћене раније у овом поглављу. Тестирање је извршено на 10 полазних великих инстанци

мет./инст.	50	70	100	120	150	200	350	500	750	1000
itcen st. n	30	33	43	63	75	94	161	218	344	461
itcen st. shuf.	19	26	36	42	48	74	115	166	252	344
itcen st. 1/C	29	31	42	62	72	89	155	213	332	449
itcen st. 1/R	29	31	41	62	72	89	154	212	331	446
itcen st. R/C	29	31	41	62	72	88	154	212	331	447
itcen din. shuf.	42	63	87	110	127	178	296	433	659	880
itcen din. 1/C	29	31	42	62	72	89	155	213	332	449
itcen din. 1/R	29	31	41	62	72	88	154	211	331	444
itcen din. R/C	29	31	41	62	72	89	154	212	331	446
bincen din. shuf.	19	28	35	45	49	76	119	171	269	363
bincen din. 1/C	29	31	42	62	72	89	155	213	332	449
bincen din. 1/R	29	31	42	62	72	89	157	214	335	462
bincen din. R/C	28	31	41	62	72	88	153	211	330	445
dotprod 1	50	70	100	120	150	200	350	500	750	1000
dotprod 2	31	45	71	80	98	125	224	335	492	657
dotprod 3	29	31	41	62	72	90	158	217	337	455
VNS-1	16	24	32	39	45	69	109	158	243	328
U-GRASP	16	24	32	39	45	71	111	160	247	334

Табела 36: Добијена решења помоћу 16 метода из рада [2], VNS-1 и U-GRASP, на инстанцама са индексом 3

од 50, 70, 100, 120, 150, 200, 350, 500, 750 и 1000 пакета. Резултати су приказани у табели 39. Свака врста у табели одговара примењеној методи, а свака колона одговара разматраној инстанци која је описана бројем пакета. Вредности у табели одговарају укупно добијеној минималној цени за превоз задатог скупа пакета. Тестирање је извршено на 10 великих инстанци из библиотеке, то је основни скуп инстанци чији индекси су 1. Овде су код метода VNS-1 и U-GRASP приказане добијене цене за 60 и 120 секунди рада по извршењу, као и раније у овом поглављу. Овде се такође добија да су методе VNS-1 и U-GRASP достигле боља решења на свим инстанцама, него методе представљене у раду [2].

У раду [2] свака од наведених метода примењена је на скупу од 4500 инстанци које су за потребе овог рада генерисане. Било је пет класа улазних података, за сваку од њих је бирана конфигурација од 10, 30 или 100 контејнера, са 2, 5 или 10 димензија и генерисано је по 100 таквих инстанци. У том раду је разматран случај када је сваки контејнер задат својим ограничењима тј. не постоје стандардни типови контејнера. Ова колекција инстанци није нигде доступна него је у раду само описан начин њиховог генерисања. Ту нема гаранције да је могуће

мет./инст.	50	70	100	120	150	200	350	500	750	1000
itcen st. n	22	33	50	51	77	87	159	243	344	452
itcen st. shuf.	16	25	38	44	51	65	120	177	250	346
itcen st. 1/C	22	32	49	49	74	85	155	238	336	443
itcen st. 1/R	21	32	49	48	73	85	154	236	334	442
itcen st. R/C	21	32	48	49	72	84	154	236	335	442
itcen din. shuf.	43	63	90	108	130	173	297	436	658	877
itcen din. 1/C	22	32	49	49	74	85	155	238	335	443
itcen din. 1/R	22	33	49	48	72	84	154	236	334	441
itcen din. R/C	21	32	49	48	72	85	154	236	334	441
bincen din. shuf.	18	26	37	47	54	69	128	183	261	364
bincen din. 1/C	22	33	49	49	74	85	155	238	336	443
bincen din. 1/R	23	34	50	49	75	87	156	238	337	443
bincen din. R/C	21	32	48	48	72	84	154	236	333	440
dotprod 1	50	70	100	120	150	200	350	500	750	1000
dotprod 2	36	40	64	84	96	137	220	313	482	646
dotprod 3	22	32	48	49	72	85	154	239	338	448
VNS-1	14	23	34	41	48	62	114	169	240	328
U-GRASP	14	23	34	42	49	63	116	172	245	333

Табела 37: Добијена решења помоћу 16 метода из рада [2], VNS-1 и U-GRASP, на инстанцама са индексом 4

пакете спаковати у расположиве контејнере, јер управо се тим проблемом одлучивања методе баве. Методе су међусобно упоређиване по томе колико су успеле да спакују инстанци, колика им је била просечна попуњеност контејнера, колико су утрошиле контејнера итд. Прва метода са скаларним производом (dotprod 1) успела је да спакује највећи број од 4500 инстанци, око 2200. Предложене методе су најјаче када се све примене на задате улазне податке и изабере најбоље од добијених решења. Евалуација њиховог рада извршена је свођењем на хомогени проблем, када су сви контејнери истих особина. Да би се одредио најмањи потребан број контејнера да се спакује задати скуп пакета у хомогеном случају коришћена је идеја бинарне претраге. Наиме, прво се одреде горња и доња граница потребног броја контејнера. Горња граница се изабере као неки довољно велики број за који бар једна од 34 методе успева да спакује пакете у тај број контејнера. Као доња граница се бира неки мањи број контејнера за који ни једна од метода не успева да спакује пакете. Потом се интервал полови и методом бинарне претраге одређује се минималан број контејнера који је потребан. За сваки изабрани број контејнера извршавају се све 34 методе и ако бар једна

мет./инст.	50	70	100	120	150	200	350	500	750	1000
itcen st. n	25	37	46	55	68	89	162	222	347	469
itcen st. shuf.	17	26	38	41	51	74	120	169	258	340
itcen st. 1/C	24	37	45	55	65	86	159	219	339	453
itcen st. 1/R	24	36	44	55	65	84	159	218	334	453
itcen st. R/C	24	35	44	54	64	84	159	217	334	453
itcen din. shuf.	42	64	87	106	127	178	294	428	665	877
itcen din. 1/C	24	37	45	55	65	86	159	219	338	453
itcen din. 1/R	24	36	45	55	64	88	159	218	336	453
itcen din. R/C	24	35	44	54	64	85	159	218	334	453
bincen din. shuf.	15	26	39	45	52	78	125	174	273	354
bincen din. 1/C	24	37	45	55	65	86	159	219	339	453
bincen din. 1/R	24	37	45	55	66	89	159	220	341	456
bincen din. R/C	23	35	45	54	64	84	158	216	333	452
dotprod 1	50	70	100	120	150	200	350	500	750	1000
dotprod 2	33	43	59	74	104	133	231	340	490	642
dotprod 3	25	35	44	55	65	84	161	220	344	463
VNS-1	14	22	36	39	47	70	114	160	246	324
U-GRASP	14	22	36	39	47	71	116	162	251	330

Табела 38: Добијена решења помоћу 16 метода из рада [2], VNS-1 и U-GRASP, на инстанцама са индексом 5

од њих успе да их спакује, сматра се да је паковање могуће па се иде на мањи број контејнера. У супротном је паковање неуспешно, па се иде на већи број контејнера. Овим приступом успешно је решено 249 од 400 стандардних хомогених инстанци уведених у раду [3]. Међутим овакав приступ са бинарном претрагом није могућ за хетерогени случај јер није јасно које контејнере разматрати када се интервал преполови. Зато предложене методе своју пуну снагу показују удружене и примењене на хомогени случај. У наредном поглављу ће методе VNS и GRASP бити приказане сведене на хомогени случај и ту ће се показати да су и тада успешније од скупа метода из рада [2].

мет./инст.	50	70	100	120	150
itcen st. n	43038	68542	90858	76512	108392
itcen st. shuf.	39987	59667	77649	87594	110607
itcen st. 1/C	39850	66948	87670	76512	103610
itcen st. 1/R	39850	66948	87670	74918	103610
itcen st. R/C	39850	66948	87670	74918	103610
itcen din. shuf.	91632	136269	192815	227328	282397
itcen din. 1/C	39850	66948	87670	76512	103610
itcen din. 1/R	39850	66948	87670	74918	103610
itcen din. R/C	39850	66948	87670	74918	102016
bincen din. shuf.	43320	58923	80329	93423	112906
bincen din. 1/C	39850	66948	87670	76512	103610
bincen din. 1/R	39850	66948	87670	74918	103610
bincen din. R/C	39850	66948	86076	74918	103610
dotprod 1	124150	173810	248300	297960	372450
dotprod 2	55393	90821	158865	183962	195034
dotprod 3	39850	66948	87670	74918	105204
VNS-1	31517	46257	63454	65906	85586
U-GRASP	32367	47996	66774	70076	91482

мет./инст.	200	350	500	750	1000
itcen st. n	146648	250258	355462	540366	720488
itcen st. shuf.	151522	267893	375833	541545	731981
itcen st. 1/C	141866	245476	345898	530802	706142
itcen st. 1/R	141866	243882	344304	527614	702954
itcen st. R/C	140272	243882	344304	527614	701360
itcen din. shuf.	377606	649563	955051	1452959	1929071
itcen din. 1/C	141866	245476	345898	530802	704548
itcen din. 1/R	140272	242288	352274	526020	699766
itcen din. R/C	140272	243882	344304	526020	702954
bincen din. shuf.	160578	282148	385174	573166	771499
bincen din. 1/C	141866	245476	345898	530802	706142
bincen din. 1/R	141866	247070	350680	530802	712518
bincen din. R/C	140272	240694	344304	524426	699766
dotprod 1	496600	869050	1241500	1862250	2483000
dotprod 2	262342	455604	640550	971771	1370577
dotprod 3	141866	247070	352274	537178	714112
VNS-1	119389	206411	292886	439640	592871
U-GRASP	128631	224475	317149	477587	641267

Табела 39: Добијене цене помоћу 16 метода из рада [2], VNS-1 и U-GRASP

5 Хомогени векторски проблем паковања контејнера

У литератури је векторски проблем паковања контејнера са два типа ограничења више разматран за хомогени случај. Тада су сви контејнери истог типа и једнаких капацитета. Када не постоји више типова контејнера, онда се за функцију циља може узети број употребљених контејнера.

Претходно разматране методе VNS и GRASP могу се свести на хомогени проблем паковања искључивањем потеза који подразумевају промену типа контејнера. Овим свођењем могуће је добити нови скуп метахеуристичких метода намењених за хомогени проблем који је у литератури посебно разматран. Овакав приступ примењен је из разлога додатне евалуације предложених метахеурситичких метода.

За хомогени векторски проблем паковања са два типа ограничења постоји стандардна библиотека од 400 инстанци уведена у раду [3]. Ове инстанце груписане су у 10 класа по 40 инстанци. У оквиру сваке класе инстанце су подељене у 4 групе, у свакој од њих је 10 инстанци са истим бројем пакета. Број пакета код свих класа је 25, 50, 100 и 200, осим код десете класе где је број пакета 24, 51, 99 и 201.

Како је овакав проблем паковања NP тежак, не постоји егзактан алгоритам који може да реши инстанце у полиномском времену. Једна од успешних метода примењена је у раду [4] из 2016. године. Приступ у том раду заснива се на свођењу проблема паковања на проток кроз граф, при чему је значајна пажња посвећена редуковању симетрије решења. На тај начин добијају се графови са мање чворова, који могу лакше да се реше. Проблем је решаван у C++, а потом помоћу решавача Gurobi. На тај начин је решено 330 инстанци, док за 70 инстанци метода из овог рада није дала решење, јер је за бар једну инстанцу из групе од њих 10 било потребно време од бар 12 сати да би се егзактно решила. Иако за многе од 330 решених инстанци примењена метода брзо долази до решења, код 10 инстанци из 1. класе величине 200 пакета, назовимо их краће 1_200, било је потребно између 5 хиљада и 19 хиљада секунди да би се решиле. Овај рад донео је значајан помак, јер за 188 инстанци од 330 решених, раније нису била позната оптимална решења. Библиотека инстанци и добијена оптимална решења методом из рада [4], могу се видети на адреси

https://research.fdabrandao.pt/research/vpsolver/results/2cbp.html

Као процена доње границе броја потребних контејнера може се користити формула

$$l_\infty = \max\left\{\left\lceil\frac{\sum m_i}{Lm}\right\rceil, \left\lceil\frac{\sum V_i}{LV}\right\rceil\right\}. \tag{5.1}$$

Ова формула је коришћена и у раду [2] за евалуацију решења на истој библиотеци од 400 инстанци. Уколико се добије број контејнера једнак вредности l_∞ онда је то гарантовано оптимално решење. У том раду [2], добија се укупно 249 оптималних решења од 400 инстанци, а од тога су 52 оптимална од 70 инстанци које нису решене у раду [4].

5.1 Решавање методом GRASP

Приликом преласка на хомогени случај занемарено је све што се у алгоритму односи на тип контејнера јер су сви контејнери истог типа. Такође у функцији циља узето је да је цена једнака 1, па се као резултат добија број употребљених контејнера.

Варијанта U-GRASP која се за хетерогени случај показала као најбоља тестирана је и за хомогени случај уз неопходне поменуте модификације. Осим тога код хетерогеног случаја у процедури GRC коришћене су различите листе кандидата, по маси или запремини, у зависности од типа контејнера. Како овде типови контејнера не фигурушу онда се сваки пут на почетку процедуре GRC на случајан начин, са подједнаком вероватноћом, бира која се од те две листе кандидата користи за иницијално паковање.

Као мера успешности методе GRASP за хомогени цлучај узет је број достигнутих оптималних решења на скупу од 400 стандардних инстанци. Додатни критеријум који је коришћен у приказивању резултата је време рада по извршењу. Добијени резултати су представљени су у раду [5], доносимо их овде у табели 40. Ту је као време рада изабрано 120 секунди. На овај начин добијено је укупно 245 оптималних решења од 400 инстанци, а од тога је 38 оптималних решења на скупу од 70 инстанци које нису биле решене у раду [4] (приказане су подебљано у табели). Повећањем времена рада са 120 на 600 секунди добијено је још 7 нових оптималних решења, укупно 252 од 400, а од тога је 2 нова од 70 нерешених, укупно 40 од 70.

класа	25/24п	50/51п	100/99п	200/201п
1	10	10	0	0
2	10	10	8	5
3	10	10	9	5
4	9	**10**	**8**	0
5	10	**10**	**10**	0
6	10	8	0	0
7	9	6	3	0
8	10	10	10	10
9	10	10	0	0
10	5	0	0	0
Σ	93	84	48	20

Табела 40: Добијена оптимална решења помоћу U-GRASP за 120s у раду [5]

класа	25/24п	50/51п	100/99п	200/201п
1	10	10	2	0
2	10	10	9	5
3	10	10	10	5
4	9	**10**	**10**	0
5	10	**10**	**10**	0
6	10	9	0	0
7	9	6	3	0
8	10	10	10	10
9	10	10	0	0
10	7	0	0	0
Σ	95	85	54	20

Табела 41: Добијена оптимална решења помоћу U-GRASP за 360s

У односу на рад [5] задржавајући исти алгоритам али уводећи неколико техничких поправки као што су смањење броја колона матрице решења бољом проценом максималног броја потребних контејнера, потом убрзавањем копирања матрица решења када се дође до дела испуњеног нулама, заменом функције за сортирање у локалној претрази, дошло је приметног убрзања налажења решења. Уз то метода је тестирана на нешто новијем рачунару бољих карактеристика, па је и то делом утицало на боље резултате од оних у раду [5]. Ефекат је такав да се за свега 1 секунду рада добија 235 оптималних решења на скупу од 400 инстанци. Повећањем времена рада редом на 10, 60 и 360 секунди број достигнутих оптималних решења порастао је редом на 245, 250 и 254. Закључује се да повећање времена рада даје боља решења али не доноси много, метода не може на тај начин да да битно боља решења од оних које постиже у првим секундама. У табели 41 дат је број оптималних решења

које метода U-GRASP достиже за 360 секунди рада по извршењу.

Са оптимално решених 254 инстанце метода је била боља од групе удружених грамзивих метода из рада [2] где је решено 249 инстанци на истом скупу. Док је у том раду решено 38 од 70 инстанци које су остале нерешене у раду [4], метода U-GRASP је решила 40 инстанци из тог скупа.

5.2 Решавање методом VNS

Као мера успешности метода VNS и GRASP узет је број достигнутих оптималних решења на скупу од 400 стандардних инстанци.

Разматрана варијанта VNS једнака је оној описаној у раду [63], као и у претходној глави које се односи на хетерогени случај. Једина разлика у односу на хомогени случај је та што су овде све активности везане за промену типа контејнера искључене, сви контејнери су истог типа, а као цена изабрана је вредност 1. Тиме се практично не мења функција циља и она у овом случају представља број употребљених контејнера. Коришћене су исте вредности параметара, $niter = 6$ и $rmax = \lfloor 0.25n \rfloor$.

класа	25/24п	50/51п	100/99п	200/201п
1	10	10	6	9
2	10	10	10	10
3	10	10	10	10
4	10	10	10	10
5	10	10	10	10
6	10	10	5	0
7	10	10	7	9
8	10	10	10	10
9	10	10	10	0
10	10	10	10	9
Σ	100	100	88	77

Табела 42: Добијена оптимална решења помоћу VNS за 1440s

Као мера успешности метода VNS и GRASP узет је број достигнутих оптималних решења на скупу од 400 стандардних инстанци. Као време рада по извршењу коришћено је 1, 10, 60, 360 и 1440 секунди. За ова времена метода је редом постигла 315, 331, 345, 356 и 365 оптималних решења на скупу од 400 инстанци. Ови резултати су такође приказани у раду [63]. Резултати рада методе VNS приказани су, за 1440 секунди рада и 30 извршавања, у табели 42. Ту се за сваку групу од по 10 инстанци наводи број достигнутих оптималних решења. Код хомогеног

проблема, где су вредности функције циља дискретне, поправке решења се ређе дешавају него када ја проблем хетероген са више цена које су реални бројеви, где су могуће чешће поправке решења.

Помоћу ове методе добијено 305 од 330 оптималних решења из рада [4]. Од преосталих 70 инстанци које тамо нису решене овде је добијено 60 оптималних решења, што даје укупан број добијених оптималних решења 365.

Осим поменуте стандардне библиотеке 400 хомегених инстанци за векторско паковање, постоји још једна новија библиотека већих 400 хомогених инстанци уведена у раду [64]. Она је добијена од претходне библиотеке с тим што се сваки пакет понавља неки број пута, а тај број је случајно изабран из интервала од 1 до 100 за сваки пакет приликом креирања инстанце. На тај начин су добијене значајно, око 50 пута, веће инстанце од полазних. Аутори у том раду наводе да су позната оптимална решења за 339 од тих 400 инстанци. За осталу 61 инстанцу оптимална решења нису позната, него се само знају границе између којих се оптимално решење налази.

Инстанца	n	LB	UB	1s	10s		60s		360s	1440s	
$CL\_04\_100\_06$	5089	627	628	691	628		**627**	*	/	/	
$CL\_04\_100\_08$	5089	642	645	687	**643**		**642**	*	/	/	
$CL\_04\_200\_01$	10229	1293	1297	1453	1373		1308		**1294**	**1293**	*
$CL\_05\_100\_06$	5089	314	315	322	**314**	*	/		/	/	
$CL\_05\_100\_08$	5089	321	323	**322**	**321**	*	/		/	/	
$CL\_05\_100\_10$	5089	327	328	333	**327**	*	/		/	/	
$CL\_05\_200\_02$	10141	624	629	682	633		**628**		**627**	**627**	
$CL\_05\_200\_03$	10157	630	635	708	646		**634**		**633**	**633**	
$CL\_05\_200\_04$	10141	630	631	670	635		631		631	**630**	*
$CL\_05\_200\_05$	10141	632	640	704	646		**634**		**633**	**632**	*
$CL\_05\_200\_06$	10141	626	630	686	630		**627**		**627**	**627**	
$CL\_05\_200\_07$	10157	630	636	715	644		636		**635**	**634**	
$CL\_05\_200\_08$	10141	635	640	681	**635**	*	/		/	/	
$CL\_05\_200\_10$	10141	632	635	690	**633**		**632**	*	/	/	

Табела 43: Поређење резултата VNS на великим инстанцама.

На тој библиотеци инстанци тестирана је метода VNS за иста времена од 1, 10, 60, 360 и 1440 секунди по извршењу. На скупу од 61 нерешене инстанце метода VNS је добила 10 гарантовано оптималних решења и код још 4 инстанце је померила (спустила) горњу границу оптималног решења. Ови резултати су представљени у раду [63], а доносимо их и у табели 43. Симболом $*$ су означена решења за инстанце код којих је

добијено гарантовано оптимално решење, а подебљано она решења која поправљају до тада познату горњу границу оптималног решења. Симбол / има значење да када је нађено оптимално решење инстанца није поново решавана за веће време рада.

5.3 Упоредни приказ решења применом CPLEX, VNS и GRASP

Ради поређења решења на свих 400 инстанци из библиотеке оне су решаване и помоћу решавача CPLEX 12.6.2 са ограничењем времена рада од 3600 секунди (1 сат). Тестирања су урађена на истом рачунару и под истим условима као и методе VNS и GRASP.

У табели 45 наведени су резултати за свих 400 инстанци. Ту свакој врсти одговара једна инстанца која се решава, док је у четири колоне приказано: оптимално решење, најбоља решења помоћу CPLEX, VNS и GRASP. Оптималност решења позната ја из литературе, пре свега за 330 инстанци позната је из рада [4], а за 60 од 70 преосталих инстанци одређена је по формули (5.1) и решена помоћу VNS. За преосталих 10 великих инстанци из групе 9_200 оптимална решења су позната на основу граница и из литературе. Приказана најбоља добијена решења помоћу метода VNS и GRASP су за максимална времена рада од 1440 и 360 секунди по извршењу.

Приликом решавања инстанци помоћу CPLEX, за неке од њих добија се оптимално решење за краће време и решавач је тада завршавао са радом. За инстанце које CPLEX не успева да реши у предвиђеном времену, наводи најбоље решење које је успео да нађе за то време. Код неких инстанци је овако пронађено најбоље решење једнако оптималном, без обзира што то CPLEX не зна. Тако добијена решења једнака оптималном су у резултатима приказана искошено јер за њих решавач не гарантује да су оптимална.

Збирни резултати рада решавача CPLEX су наведени у табели 44. Ту сваки ред одговара једној од десет класа, а колоне одговарају величинама инстанци, као резултат приказан је број добијених оптималних решења на тој групи од 10 инстанци, без обзира да ли је CPLEX успео да докаже њихову оптималност. Може се приметити да је CPLEX за време рада од 3600 секунди по инстанци достигао 310 оптималних ре-

класа	25/24п	50/51п	100/99п	200/201п
1	10	10	5	0
2	10	10	10	10
3	10	10	10	10
4	10	10	10	3
5	10	10	10	10
6	10	10	5	1
7	10	8	6	0
8	10	10	10	10
9	10	10	5	0
10	10	7	0	0
Σ	100	95	71	44

Табела 44: Достигнута оптимална решења помоћу CPLEX за 3600s

шења, од тога за 63 њих није гарантовао да су оптимална и радио их је до истека времена. Поредећи решења добијена помоћу CPLEX и помоћу VNS може се приметити значајна разлика у укупном броју оптимално решених инстанци. Предности методе VNS су те да за врло кратко време добија висок квалитет решења, док CPLEX гарантује за квалитет добијених решења. Осим тога, може се закључити да је предложена VNS метода за хетерогени случај приликом преласка на хомогени случај тестирањем на стандардној библиотеци од 400 инстанци показала добар ниво квалитета упоредив или бољи од оних који се могу добити помоћу решавача, или су представљени у литератури. Такође, повећањем броја пакета метахеуристичке методе показују предност у квалитету добијених решења у односу на решаваче, што је посебно видљиво за нешто већи број пакета од оних у доступној библиотеци за хомогени случај, где је највећа величина инстанце 201 пакет.

Укупно, могу се упоредити добијени резултати на библиотеци од 400 инстанци уведеној у раду [3]. За поређење се може користити рад [4] у којем је оптимално решено 330 инстанци, док за 70 инстанци нису добијена решења. Овај рад је узет као репер јер је донео значајан помак, за 188 инстанци од 330 решених, раније нису била позната оптимална решења. У раду [2] добијено је 249 оптималних решења на скупу од 400 инстанци. Од тога су 52 нова оптимална решења на скупу од 70 инстанци које нису решене у раду [4], а чија је оптималност доказана коришћењем формуле (5.1). Овде је приказана варијанта методе GRASP постигла 254 оптимална решења на скупу од 400 инстанци, од тога су 40 оптимална решења на скупу од 70 инстанци које нису решене у раду [4]. Затим, овде најбоља приказана варијанта методе VNS постиже 365 оптималних

решења на скупу од 400 инстанци, од тога је 60 оптималнуих решења на скупу 70 инстанци које нису решене у раду [4]. VNS је радио највише 1440 секунди по извршењу, и понављао рад 30 пута. На крају је ради верификације коришћен решавач CPLEX који је под истим условима добио сат времена за рад и добио је гарантовано оптималних 247 решења на 400 инстанци, од чега су 53 биле из скупа 70 инстанци које нису решене у раду [4]. Осим тога CPLEX је и за остале 153 инстанце приказао решења која је добио, али за која не гарантује да су оптимална. На том скупу је добио решења за још 63 инстанце чија оптималност је позната из других извора.

инстанца	опт	CPLEX	VNS	GRASP	инстанца	опт	CPLEX	VNS	GRASP
1_25_1	6	6	6	6	2_25_1	13	13	13	13
1_25_2	7	7	7	7	2_25_2	14	14	14	14
1_25_3	7	7	7	7	2_25_3	14	14	14	14
1_25_4	7	7	7	7	2_25_4	14	14	14	14
1_25_5	7	7	7	7	2_25_5	13	13	13	13
1_25_6	7	7	7	7	2_25_6	14	14	14	14
1_25_7	7	7	7	7	2_25_7	14	14	14	14
1_25_8	7	7	7	7	2_25_8	15	15	15	15
1_25_9	7	7	7	7	2_25_9	15	15	15	15
1_25_10	7	7	7	7	2_25_10	16	16	16	16
1_50_1	13	13	13	13	2_50_1	30	30	30	30
1_50_2	13	13	13	13	2_50_2	31	31	31	31
1_50_3	13	13	13	13	2_50_3	31	31	31	31
1_50_4	13	13	13	13	2_50_4	31	31	31	31
1_50_5	13	13	13	13	2_50_5	31	31	31	31
1_50_6	14	14	14	14	2_50_6	32	32	32	32
1_50_7	14	14	14	14	2_50_7	32	32	32	32
1_50_8	14	14	14	14	2_50_8	32	32	32	32
1_50_9	14	14	14	14	2_50_9	33	33	33	33
1_50_10	14	14	14	14	2_50_10	32	32	32	32
1_100_1	25	26	26	26	2_100_1	62	62	62	63
1_100_2	26	26	26	26	2_100_2	57	*57*	57	57
1_100_3	26	26	26	26	2_100_3	56	56	56	56
1_100_4	25	26	25	26	2_100_4	57	*57*	57	57
1_100_5	25	26	26	26	2_100_5	56	56	56	56
1_100_6	25	26	26	26	2_100_6	57	*57*	57	57
1_100_7	25	26	26	27	2_100_7	56	56	56	56
1_100_8	26	26	26	27	2_100_8	58	*58*	58	58
1_100_9	26	26	26	27	2_100_9	57	57	57	57
1_100_10	26	26	26	27	2_100_10	58	*58*	58	58
1_200_1	50	53	50	54	2_200_1	111	*111*	111	113
1_200_2	50	53	50	53	2_200_2	114	*114*	114	114
1_200_3	50	52	50	54	2_200_3	111	*111*	111	113
1_200_4	50	52	50	54	2_200_4	115	*115*	115	115
1_200_5	50	52	50	54	2_200_5	110	*110*	110	112
1_200_6	50	53	50	54	2_200_6	116	*116*	116	116
1_200_7	50	53	51	54	2_200_7	111	*111*	111	113
1_200_8	51	52	51	54	2_200_8	117	*117*	117	117
1_200_9	51	52	51	54	2_200_9	112	*112*	112	114
1_200_10	51	53	51	54	2_200_10	118	*118*	118	118

Табела 45: Добијена решења помоћу CPLEX, VNS и GRASP

инстанца	опт	CPLEX	VNS	GRASP	инстанца	опт	CPLEX	VNS	GRASP
3_25_1	13	13	13	13	4_25_1	3	3	3	3
3_25_2	14	14	14	14	4_25_2	3	3	3	3
3_25_3	14	14	14	14	4_25_3	3	3	3	3
3_25_4	14	14	14	14	4_25_4	3	3	3	4
3_25_5	13	13	13	13	4_25_5	3	3	3	3
3_25_6	14	14	14	14	4_25_6	3	3	3	3
3_25_7	14	14	14	14	4_25_7	3	3	3	3
3_25_8	15	15	15	15	4_25_8	4	4	4	4
3_25_9	15	15	15	15	4_25_9	4	4	4	4
3_25_10	16	16	16	16	4_25_10	4	4	4	4
3_50_1	30	30	30	30	4_50_1	7	7	7	7
3_50_2	31	31	31	31	4_50_2	7	7	7	7
3_50_3	31	31	31	31	4_50_3	7	7	7	7
3_50_4	31	31	31	31	4_50_4	7	7	7	7
3_50_5	31	31	31	31	4_50_5	7	7	7	7
3_50_6	32	32	32	32	4_50_6	7	7	7	7
3_50_7	32	32	32	32	4_50_7	7	7	7	7
3_50_8	32	32	32	32	4_50_8	7	7	7	7
3_50_9	33	33	33	33	4_50_9	7	7	7	7
3_50_10	32	32	32	32	4_50_10	7	7	7	7
3_100_1	56	56	56	56	4_100_1	13	13	13	13
3_100_2	57	*57*	57	57	4_100_2	13	13	13	13
3_100_3	57	57	57	57	4_100_3	13	13	13	13
3_100_4	57	*57*	57	57	4_100_4	13	13	13	13
3_100_5	56	56	56	56	4_100_5	13	13	13	13
3_100_6	57	*57*	57	57	4_100_6	13	13	13	13
3_100_7	56	56	56	56	4_100_7	13	13	13	13
3_100_8	58	*58*	58	58	4_100_8	13	13	13	13
3_100_9	57	57	57	57	4_100_9	13	13	13	13
3_100_10	58	*58*	58	58	4_100_10	13	13	13	13
3_200_1	111	*111*	111	113	4_200_1	25	26	25	27
3_200_2	114	*114*	114	114	4_200_2	25	26	25	27
3_200_3	111	*111*	111	113	4_200_3	25	26	25	27
3_200_4	115	*115*	115	115	4_200_4	25	26	25	27
3_200_5	110	110	110	112	4_200_5	25	26	25	27
3_200_6	116	*116*	116	116	4_200_6	25	26	25	27
3_200_7	111	*111*	111	113	4_200_7	25	26	25	27
3_200_8	117	*117*	117	117	4_200_8	26	26	26	27
3_200_9	112	112	112	114	4_200_9	26	26	26	27
3_200_10	118	*118*	118	118	4_200_10	26	26	26	27

Табела 45: (наставак) Добијена решења помоћу CPLEX, VNS и GRASP

инстанца	опт	CPLEX	VNS	GRASP	инстанца	опт	CPLEX	VNS	GRASP
5_25_1	2	2	2	2	6_25_1	10	10	10	10
5_25_2	2	2	2	2	6_25_2	10	10	10	10
5_25_3	2	2	2	2	6_25_3	10	10	10	10
5_25_4	2	2	2	2	6_25_4	10	10	10	10
5_25_5	2	2	2	2	6_25_5	10	10	10	10
5_25_6	2	2	2	2	6_25_6	10	10	10	10
5_25_7	2	2	2	2	6_25_7	10	10	10	10
5_25_8	2	2	2	2	6_25_8	10	10	10	10
5_25_9	2	2	2	2	6_25_9	10	10	10	10
5_25_10	2	2	2	2	6_25_10	11	11	11	11
5_50_1	4	4	4	4	6_50_1	21	21	21	21
5_50_2	4	4	4	4	6_50_2	21	21	21	22
5_50_3	4	4	4	4	6_50_3	21	21	21	21
5_50_4	4	4	4	4	6_50_4	21	21	21	21
5_50_5	4	4	4	4	6_50_5	21	21	21	21
5_50_6	4	4	4	4	6_50_6	22	*22*	22	22
5_50_7	4	4	4	4	6_50_7	22	*22*	22	22
5_50_8	4	4	4	4	6_50_8	22	22	22	22
5_50_9	4	4	4	4	6_50_9	22	22	22	22
5_50_10	4	4	4	4	6_50_10	22	22	22	22
5_100_1	7	7	7	7	6_100_1	41	41	42	43
5_100_2	7	7	7	7	6_100_2	41	42	41	43
5_100_3	7	7	7	7	6_100_3	41	41	42	43
5_100_4	7	7	7	7	6_100_4	41	42	41	42
5_100_5	7	7	7	7	6_100_5	41	41	41	43
5_100_6	7	7	7	7	6_100_6	41	42	41	42
5_100_7	7	7	7	7	6_100_7	41	41	42	43
5_100_8	7	7	7	7	6_100_8	41	42	41	43
5_100_9	7	7	7	7	6_100_9	41	41	42	43
5_100_10	7	7	7	7	6_100_10	41	42	42	43
5_200_1	13	13	13	14	6_200_1	81	82	82	86
5_200_2	13	13	13	14	6_200_2	81	83	83	85
5_200_3	13	13	13	14	6_200_3	81	82	83	86
5_200_4	13	13	13	14	6_200_4	81	84	83	86
5_200_5	13	13	13	14	6_200_5	81	*81*	82	85
5_200_6	13	13	13	14	6_200_6	81	84	83	85
5_200_7	13	13	13	14	6_200_7	81	82	83	86
5_200_8	13	13	13	14	6_200_8	81	84	83	85
5_200_9	13	13	13	14	6_200_9	81	82	83	86
5_200_10	13	13	13	14	6_200_10	82	85	83	86

Табела 45: (наставак) Добијена решења помоћу CPLEX, VNS и GRASP

инстанца	опт	CPLEX	VNS	GRASP	инстанца	опт	CPLEX	VNS	GRASP
7_25_1	9	9	9	9	8_25_1	13	13	13	13
7_25_2	9	9	9	10	8_25_2	13	13	13	13
7_25_3	10	10	10	10	8_25_3	13	13	13	13
7_25_4	10	10	10	10	8_25_4	13	13	13	13
7_25_5	10	10	10	10	8_25_5	13	13	13	13
7_25_6	10	10	10	10	8_25_6	13	13	13	13
7_25_7	10	10	10	10	8_25_7	13	13	13	13
7_25_8	9	9	9	9	8_25_8	13	13	13	13
7_25_9	10	10	10	10	8_25_9	13	13	13	13
7_25_10	9	9	9	9	8_25_10	13	13	13	13
7_50_1	21	21	21	21	8_50_1	25	25	25	25
7_50_2	18	19	18	19	8_50_2	25	25	25	25
7_50_3	21	21	21	21	8_50_3	25	25	25	25
7_50_4	18	19	18	19	8_50_4	25	*25*	25	25
7_50_5	21	21	21	21	8_50_5	25	25	25	25
7_50_6	18	18	18	19	8_50_6	25	25	25	25
7_50_7	22	*22*	22	22	8_50_7	25	25	25	25
7_50_8	18	18	18	18	8_50_8	25	25	25	25
7_50_9	22	22	22	22	8_50_9	25	25	25	25
7_50_10	18	18	18	19	8_50_10	25	25	25	25
7_100_1	41	*41*	41	41	8_100_1	50	*50*	50	50
7_100_2	39	40	40	40	8_100_2	50	*50*	50	50
7_100_3	41	41	41	42	8_100_3	50	*50*	50	50
7_100_4	39	40	40	40	8_100_4	50	*50*	50	50
7_100_5	41	*41*	41	41	8_100_5	50	*50*	50	50
7_100_6	39	40	40	40	8_100_6	50	*50*	50	50
7_100_7	41	*41*	41	42	8_100_7	50	*50*	50	50
7_100_8	40	*40*	40	40	8_100_8	50	50	50	50
7_100_9	41	41	41	42	8_100_9	50	*50*	50	50
7_100_10	40	41	40	41	8_100_10	50	*50*	50	50
7_200_1	80	82	80	82	8_200_1	100	*100*	100	100
7_200_2	80	81	80	82	8_200_2	100	*100*	100	100
7_200_3	80	81	80	82	8_200_3	100	*100*	100	100
7_200_4	80	81	80	82	8_200_4	100	100	100	100
7_200_5	80	82	80	82	8_200_5	100	100	100	100
7_200_6	80	81	80	81	8_200_6	100	100	100	100
7_200_7	80	82	81	83	8_200_7	100	*100*	100	100
7_200_8	80	81	80	81	8_200_8	100	100	100	100
7_200_9	81	83	81	83	8_200_9	100	*100*	100	100
7_200_10	80	82	80	82	8_200_10	100	*100*	100	100

Табела 45: (наставак) Добијена решења помоћу CPLEX, VNS и GRASP

инстанца	опт	CPLEX	VNS	GRASP	инстанца	опт	CPLEX	VNS	GRASP
9_25_1	7	7	7	7	10_24_1	8	8	8	8
9_25_2	7	7	7	7	10_24_2	8	8	8	8
9_25_3	7	7	7	7	10_24_3	8	8	8	8
9_25_4	7	7	7	7	10_24_4	8	8	8	9
9_25_5	7	7	7	7	10_24_5	8	8	8	8
9_25_6	7	7	7	7	10_24_6	8	8	8	9
9_25_7	7	7	7	7	10_24_7	8	8	8	8
9_25_8	8	8	8	8	10_24_8	8	8	8	8
9_25_9	8	8	8	8	10_24_9	8	8	8	8
9_25_10	8	8	8	8	10_24_10	8	8	8	9
9_50_1	14	*14*	14	14	10_51_1	17	17	17	18
9_50_2	14	*14*	14	14	10_51_2	17	18	17	18
9_50_3	14	*14*	14	14	10_51_3	17	17	17	18
9_50_4	14	*14*	14	14	10_51_4	17	17	17	18
9_50_5	14	14	14	14	10_51_5	17	18	17	18
9_50_6	15	15	15	15	10_51_6	17	17	17	18
9_50_7	15	*15*	15	15	10_51_7	17	18	17	18
9_50_8	15	15	15	15	10_51_8	17	17	17	18
9_50_9	15	15	15	15	10_51_9	17	17	17	18
9_50_10	15	*15*	15	15	10_51_10	17	17	17	18
9_100_1	26	27	26	27	10_99_1	33	35	33	35
9_100_2	27	*27*	27	28	10_99_2	33	34	33	35
9_100_3	27	28	27	28	10_99_3	33	34	33	36
9_100_4	26	27	26	27	10_99_4	33	34	33	36
9_100_5	26	27	26	27	10_99_5	33	34	33	36
9_100_6	27	*27*	27	28	10_99_6	33	34	33	35
9_100_7	27	28	27	28	10_99_7	33	34	33	36
9_100_8	27	*27*	27	28	10_99_8	33	34	33	36
9_100_9	27	*27*	27	28	10_99_9	33	34	33	35
9_100_10	27	*27*	27	28	10_99_10	33	34	33	35
9_200_1	50	52	51	54	10_201_1	67	70	67	72
9_200_2	50	53	51	54	10_201_2	67	70	68	72
9_200_3	50	53	51	54	10_201_3	67	69	67	72
9_200_4	50	53	51	54	10_201_4	67	69	67	72
9_200_5	50	53	51	54	10_201_5	67	69	67	72
9_200_6	50	53	51	54	10_201_6	67	70	67	72
9_200_7	50	53	51	54	10_201_7	67	70	67	72
9_200_8	51	53	52	55	10_201_8	67	70	67	72
9_200_9	51	54	52	55	10_201_9	67	70	67	72
9_200_10	51	55	52	55	10_201_10	67	70	67	72

Табела 45: (наставак) Добијена решења помоћу CPLEX, VNS и GRASP

6 Закључак

У овом раду разматрани су проблеми оптимизације везани за интермодални транспорт. Приликом оптимизације разматране су вредности параметара: цена, време и загађење. Направљени су одговарајући математички модели који су затим рачунарски имплементирани и на тај начин су одређивана оптимална решења. Оптимална решења могу да се односе на тражење оптималне руте за превоз одређених типова контејнера по једном или више критеријума. У 3. глави се приликом оптимизације разматрају велики пакети који могу да заузимају више контејнера. Комбиновањем могућих расподела пакета са техникама оптимизације за превоз контејнера, добијају се решења целог проблема. У 4. и 5. глави су разматрани могући случајеви оптимизације повезани са паковањем у којима се, осим контејнера, разматра паковање више пакета. У питању су мали пакети такви да може више да их стане у један контејнер. Код ових проблема узимане су у обзир маса и запремина задатих пакета, као и ограничења масе и запремине самих контејнера. Разматрани проблем паковања се убраја у векторско паковање, при чему је у 4. глави разматран хетерогени, а у 5. глави хомогени случај. Приступ да се користе маса и запремина пакета проистекао је из праксе компанија које се баве контејнерским транспортом, као и из стандарда који дефинишу ова ограничења.

У циљу решавања проблема интермодалног транспорта креирано је више математичких модела. Представљено је и имплементирано налажење Парето скупа приликом оптимизације по два циља, цени и загађењу. За решавање модела коришћени су егзактни решавачи, CPLEX и у мањој мери Lingo. Код хетерогеног и хомогеног проблема паковања у збирном контејнерском транспорту коришћен је познати математички модел векторског паковања са два типа ограничења. Имајући у виду да егзактни решавачи не могу ефикасно да реше проблем оптимизације за велике инстанце, за проблеме паковања предложени су метахеуристички алгоритми. Креирано је више варијанти метахеуристичких метода GRASP и VNS за хетерогени и хомогени проблем паковања. Овим методама се добијају приближна решења за велике инстанце, али за знатно краће време. На малим инстанцама ова решења се, очекивано, поклапају са оптималним. За тестирање алогоритама хетерогеног паковања креирана је сопствена библиотека од 50 великих и 6 малих инстанци. Тестирање

алогоритама хомогеног паковања реализовано је на познатој библиотеци од 400 инстанци описаној у раду [3] и на новијој библиотеци 400 већих инстанци описаној у раду [64].

Главни научни допринос овог рада огледа се у следећем:

- Формирано је више математичких модела представљених у 3. поглављу, који описују разне аспекте интермодалног транспорта. У моделима се разматрају контејнери, а потом се укључују и велики пакети који се распоређују у контејнере, а задати су својим масама и запреминама. Део ових резултата објављен је у радовима [30] и [42].

- Развијено је више варијанти метахуристичких метода GRASP и VNS у области паковања збирног контејнерског транспорта, које помажу да се велики број пакета ефикасно распореди на контејнере у складу са задатом циљном функцијом. Имплементиране су у програмском језику C. Посебно су решавани хетерогени и хомогени проблем векторског паковања.

  - За хетерогени случај генерисана је сопствена библиотека од 50 инстанци, по 5 примерака у свакој од следећих 10 величина: 50, 70, 100, 120, 150, 200, 350, 500, 750 и 1000 пакета. Осим тога генерисано је и 6 малих инстанци, величина: 10, 11, 12, 13, 15 и 20 пакета. Добијена решења су тестирана на овој библиотеци инстанци, потом су упоређивана међусобно и са оним решењима која је добио егзактни решавач CPLEX. Осим тога варијанте метода GRASP и VNS су упоређене са оним приказаним у раду [2] и показале су боље перформансе приликом одређивања минималног броја употребљених контејнера, као и код одређивања минималне цене транспорта свих пакета. Резултати VNS објављени су у раду [63], а за GRASP у раду [5].
  - За хомогени случај добијена решења су тестирана на познатој библиотеци од 400 инстанци из рада [3] и упоређена су са решењима из литературе, међусобно, као и са решењима која је добио егзактни решавач CPLEX. Ови резултати за методу GRASP објављени су у раду [5], при чему су овде додатно поправљени. Најбољи резултат постигла је варијанта методе VNS која је на библиотеци од 400 хомогених инстанци омогућила проналажење укупно 365 гарантовано оптималних решења. При томе је опти-

мално решено 60 од 70 нерешених инстанци из рада [4]. Осим поређења са тим радом, урађено је поређење и са радом [2] и другима за хомогени случај паковања, и добијени су бољи резултати. Тако је на скупу од 400 инстанци предложена метода VNS решила 365 инстанци, док је у радовима [4], [2] и [64] решено редом 330, 249 и 370 инстанци. Код рада где је решен већи број инстанци, на појединим теже решивим подскуповима инстанци предложена метода VNS добила је више оптималних решења. Ови резултати су објављени у раду [63].

– Резултат који завређује највише пажње је прво налажење оптималних решења за 10 великих хомогених инстанци први пут описаних у раду [64] из 2018. године. До ових резултата у тој библиотеци постојала је 61 нерешена инстанца од постојећих 400. Осим 10 нових оптималних решења за још четири велике нерешене инстанце из исте библиотеке спуштена је горња граница оптималног решења. Ови резултати су објављени у раду [63].

Разматрана проблематика је актуелна и налази своју примену у пракси. У области збирног контејнерског транспорта могућа су усложњавања модела где пакети не би морали да полазе са једног одредишта, него из више њих. Други могући аспект примене ових истраживања јесте у самом стратешком планирању контејнерског саобраћаја како би се смањили укупни трошкови превоза и емисија угљен-диоксида на глобалном нивоу.

Додатак А Примери пакета

У поглављима 4 и 5 о паковању и збирном контејнерском транспорту коришћене су инстанце од 10, 11, 12, 13, 15, 20, 50, 70, 100, 120, 150, 200, 350, 500, 750 и 1000 пакета чије су масе и запремине случајно генерисане. Ради лакше анализе добијених резултата, као и њихове проверљивости, у овом додатку се доносе подаци који су коришћени као улазне инстанце. Инстанца од 10 пакета је већ наведена у табели 10 па је не доносимо поново. Код свих инстанци масе m су изражене у тонама, а запремине V у кубним метрима, па се то не наглашава у табелама које следе.

бр.	1	2	3	4	5	6	7	8	9	10	11
m	8	4	9	12	15	10	1	6	10	12	2
V	17	17	22	4	8	22	2	13	7	20	2

Табела 46: Инстанца од 11 пакета

бр.	1	2	3	4	5	6	7	8	9	10	11	12
m	13	3	5	1	10	9	12	14	9	12	13	9
V	20	2	5	19	11	13	23	12	12	5	4	16

Табела 47: Инстанца од 12 пакета

бр.	1	2	3	4	5	6	7	8	9	10	11	12	13
m	10	12	13	13	1	2	10	14	2	6	4	15	12
V	18	6	14	13	20	21	5	17	24	19	3	15	2

Табела 48: Инстанца од 13 пакета

бр.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
m	6	12	7	1	11	14	5	8	13	15	12	12	7	15	8
V	21	7	21	4	3	17	22	14	16	10	21	21	10	24	12

Табела 49: Инстанца од 15 пакета

бр.	1	2	3	4	5	6	7	8	9	10
m	8	15	9	1	5	2	12	8	5	10
V	8	15	19	12	16	19	18	14	15	5
бр.	11	12	13	14	15	16	17	18	19	20
m	5	14	8	9	8	5	15	2	1	7
V	17	25	5	18	13	17	10	14	8	23

Табела 50: Инстанца од 20 пакета

m	V	m	V	m	V	m	V	m	V
14	1	14	22	1	20	4	3	6	18
5	22	8	21	14	10	7	23	11	25
7	21	14	6	8	5	1	1	2	3
15	25	5	25	4	14	7	3	12	3
13	14	2	11	3	4	12	19	10	24
5	9	13	19	12	8	4	12	13	9
12	25	3	8	2	17	13	23	10	20
3	17	14	1	1	6	8	22	14	17
12	9	4	24	12	22	5	13	7	14
8	6	8	11	1	18	6	9	14	5

Табела 51: Инстанца од 50 пакета

m	V	m	V	m	V	m	V	m	V	m	V	m	V
12	10	2	16	12	20	10	15	10	19	2	22	8	19
15	2	11	18	6	22	11	13	4	19	3	25	4	17
10	25	14	2	3	18	3	11	6	16	2	17	2	16
3	18	12	18	13	14	11	8	1	8	11	2	4	13
15	19	10	20	6	23	8	23	8	23	4	21	11	13
12	8	14	18	9	12	5	3	15	14	14	20	2	10
13	24	7	7	3	3	9	11	15	21	10	24	9	23
14	2	15	3	2	25	15	25	7	19	9	9	12	9
9	15	8	14	6	9	10	15	11	16	2	25	15	21
10	21	2	5	3	10	2	18	6	16	3	21	4	14

Табела 52: Инстанца од 70 пакета

m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V
5	16	15	21	14	18	1	13	1	23	7	24	7	21	7	8	4	25	4	10
5	22	15	13	14	23	11	24	1	9	9	18	10	13	4	12	14	9	10	12
3	21	13	6	5	15	9	18	12	5	7	12	10	9	2	7	10	14	14	24
12	16	2	7	3	13	15	14	3	11	9	17	3	11	11	2	9	3	7	5
13	8	2	9	2	11	15	23	7	6	7	20	3	17	10	23	5	21	11	24
9	19	4	22	2	14	10	10	13	11	14	16	8	21	4	13	7	25	11	22
3	24	11	1	6	17	6	12	14	25	4	24	10	15	2	21	3	5	15	13
7	23	9	8	4	22	13	14	4	22	6	24	3	17	3	10	1	25	11	14
5	20	11	18	11	19	15	22	11	10	11	18	1	19	4	6	6	16	15	4
6	15	5	25	15	9	2	1	4	13	12	9	10	21	2	7	15	13	2	15

Табела 53: Инстанца од 100 пакета

m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V
1	8	4	10	11	20	2	1	11	10	5	8	4	9	7	21	11	23	9	20
9	3	14	9	5	23	15	11	11	21	4	22	9	6	8	10	2	25	7	8
10	4	10	5	12	6	1	2	1	6	6	15	6	7	6	12	15	20	2	8
11	19	14	14	2	13	4	18	1	14	4	18	6	1	15	15	9	6	4	23
14	10	3	12	10	8	2	11	11	22	11	9	10	5	15	19	9	22	15	3
10	6	2	23	8	11	8	3	10	21	13	15	1	8	3	14	9	2	1	8
2	1	10	18	3	14	3	23	8	2	4	20	13	7	11	7	11	4	2	7
5	12	8	18	3	20	4	1	1	7	4	5	2	7	4	5	10	8	14	12
6	10	2	5	2	3	11	12	3	20	9	10	2	16	5	12	9	2	13	24
4	7	11	1	13	4	13	1	9	3	13	21	5	22	14	9	11	16	13	10
7	15	15	7	8	23	15	7	9	4	5	23	9	10	12	21	6	13	15	7
6	20	15	7	8	25	3	14	8	1	6	21	4	2	14	8	13	3	12	8

Табела 54: Инстанца од 120 пакета

m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V
10	23	12	11	7	6	9	19	10	21	7	9	14	15	12	18	14	10	4	9
6	1	12	14	10	8	3	4	5	13	12	1	13	11	3	15	14	16	9	18
11	3	4	16	4	6	1	3	5	11	3	15	14	24	3	22	2	4	12	16
12	1	10	15	13	24	6	10	8	14	7	13	5	6	3	7	5	13	3	10
10	3	10	24	5	12	2	20	3	4	14	8	14	22	3	10	5	19	12	21
2	21	12	4	1	21	3	21	15	13	2	22	13	15	5	5	6	16	8	15
3	24	1	7	12	14	5	3	7	13	3	4	5	25	1	9	2	5	14	23
12	8	8	13	5	24	15	24	12	12	14	21	1	9	15	24	8	19	4	13
10	16	3	6	3	13	11	3	8	3	1	13	10	22	12	2	13	8	12	16
5	4	4	1	7	18	1	23	5	17	9	13	13	5	3	5	3	14	3	1
3	15	7	7	13	1	11	15	13	4	5	22	14	7	13	3	6	12	4	8
3	13	12	22	12	20	9	24	1	6	1	4	12	10	12	19	8	10	4	12
2	11	12	3	4	4	13	8	7	17	2	20	2	12	3	20	11	15	14	14
6	17	1	4	7	13	11	22	12	7	4	4	8	9	1	24	10	19	1	21
2	2	13	3	14	2	12	25	5	21	9	14	8	20	15	1	10	23	15	6

Табела 55: Инстанца од 150 пакета

m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V
14	19	9	6	3	17	10	7	3	24	15	4	6	14	12	7	6	9	10	10
10	15	3	12	5	2	11	8	6	21	1	8	6	11	5	3	9	1	10	8
14	11	9	18	10	17	6	7	6	20	9	11	8	5	4	2	14	21	10	16
10	9	11	11	6	24	4	14	4	11	3	17	5	9	4	19	3	15	4	25
8	10	12	3	10	17	2	15	13	8	11	21	4	18	5	2	5	10	13	6
12	9	13	18	4	6	15	6	9	21	3	19	3	4	9	21	3	22	4	24
15	2	3	13	2	3	9	17	8	12	12	23	8	20	6	12	9	18	8	16
10	16	3	14	14	2	5	5	11	2	6	8	2	17	7	16	10	23	4	14
5	5	12	17	8	10	6	24	12	7	10	20	11	18	9	23	3	5	11	10
8	19	15	10	13	8	2	25	9	5	13	15	9	13	14	18	6	13	14	4
9	21	15	21	11	25	6	25	10	7	13	24	15	17	10	8	5	13	5	15
7	5	12	3	4	25	6	6	15	12	14	5	8	12	8	25	11	2	2	9
2	16	9	6	8	23	1	5	11	18	9	4	9	23	8	21	14	14	13	25
6	23	1	12	2	21	15	11	4	5	4	24	10	14	15	14	14	2	2	18
5	12	15	19	3	1	5	10	14	19	8	22	7	2	5	13	3	5	15	10
6	20	12	20	2	20	9	13	6	15	15	19	1	11	4	14	2	12	10	20
4	16	13	2	12	6	7	22	13	11	3	6	2	8	5	16	3	5	9	11
13	5	13	16	6	17	9	12	3	24	4	18	2	1	14	6	12	12	15	14
14	23	1	1	10	23	1	6	4	1	10	5	9	10	15	4	6	2	3	7
11	21	9	13	7	3	12	11	4	19	4	18	14	20	8	12	12	18	10	19

Табела 56: Инстанца од 200 пакета

m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V
5	9	5	9	8	22	5	3	1	21	6	7	15	23	8	17	1	7	8	2
12	22	6	3	2	25	4	15	12	9	11	17	15	17	14	24	7	21	14	9
10	5	15	21	1	11	8	1	1	15	6	12	8	13	1	17	3	18	11	23
9	22	12	12	9	20	15	1	13	6	8	20	14	23	4	11	11	16	7	18
4	10	5	1	13	5	14	4	12	19	4	25	11	20	11	11	11	23	13	4
2	22	15	20	8	23	11	5	5	21	13	12	13	18	13	17	5	14	13	15
6	22	6	2	13	12	7	10	1	1	14	23	11	16	10	18	4	7	9	7
15	15	11	16	15	8	7	3	1	4	9	5	2	19	15	22	13	22	12	2
14	19	10	19	11	12	11	9	3	10	10	1	12	6	4	14	6	12	1	10
14	2	5	19	2	2	3	7	2	20	1	9	2	3	14	3	10	6	11	21
3	20	13	24	9	19	3	8	1	7	9	13	1	18	10	4	2	23	4	14
5	22	14	11	2	9	15	7	7	11	15	12	4	25	12	17	7	12	10	7
14	7	1	7	11	11	7	5	12	8	15	22	14	15	3	13	14	17	2	12
9	23	2	16	2	25	10	11	14	21	14	11	7	5	1	23	4	18	12	25
11	14	13	17	7	6	4	7	5	1	8	2	9	19	12	8	13	22	7	1
10	13	5	17	10	14	3	9	14	5	4	11	3	25	14	21	15	5	15	24
6	25	11	17	7	19	3	2	15	4	12	4	4	25	12	22	5	14	5	25
1	8	5	13	2	12	3	23	7	23	4	11	8	12	2	9	8	4	7	18
4	20	10	18	10	14	8	9	5	1	4	7	11	15	15	25	3	5	15	3
13	2	15	7	2	15	12	10	15	14	10	5	7	7	1	19	3	20	5	24
5	23	12	19	15	9	9	7	8	21	11	23	7	2	11	5	12	7	9	17
2	10	14	9	6	15	12	14	15	12	3	23	3	12	11	22	2	17	3	12
13	12	4	11	11	9	10	14	7	19	8	13	12	7	7	15	13	15	7	25
8	20	15	22	13	19	10	2	13	13	9	10	2	23	3	5	13	2	2	13
15	16	7	6	7	25	8	7	4	25	4	13	7	4	1	4	3	5	10	1
5	18	3	23	15	1	11	12	1	18	3	8	3	10	8	12	15	12	10	2
13	7	13	9	14	15	7	25	2	2	11	16	5	10	14	9	6	17	5	21
8	12	7	12	14	4	11	3	10	23	8	4	15	16	2	5	12	9	10	6
10	16	1	11	9	14	14	10	9	1	10	22	9	5	3	19	8	23	1	18
8	23	12	20	9	11	6	6	10	2	9	1	7	25	13	4	4	10	8	6
10	17	13	17	7	25	2	11	13	2	15	14	9	9	7	3	2	11	9	25
11	18	8	1	9	5	2	4	1	24	1	9	15	1	13	7	2	18	5	23
11	8	11	16	14	6	12	9	3	18	6	8	3	1	2	8	1	9	10	8
10	9	13	15	11	21	10	2	4	1	4	24	5	11	1	23	6	23	1	9
3	15	7	4	9	20	3	15	4	6	14	5	10	12	13	9	7	13	10	21

Табела 57: Инстанца од 350 пакета

m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V
12	24	13	4	5	23	5	15	1	1	5	2	2	4	8	7	6	18	10	8
11	21	10	2	9	13	14	6	10	4	1	11	13	9	4	20	14	13	9	24
1	12	3	10	2	23	3	6	1	11	8	5	12	7	9	7	7	9	11	9
3	6	8	4	13	16	5	21	14	20	6	21	2	19	8	12	2	25	14	7
8	21	11	21	13	4	8	19	9	16	12	24	13	8	14	18	1	25	12	19
13	24	8	3	12	4	3	19	13	7	7	8	10	12	12	7	5	3	10	18
3	22	12	22	12	8	8	1	9	20	3	17	10	24	11	23	5	9	1	2
15	1	12	21	11	1	6	16	4	21	15	7	12	10	7	1	14	6	11	22
5	14	12	11	13	22	14	10	4	19	12	5	4	15	11	16	11	25	8	2
5	25	2	9	4	11	14	4	4	16	3	15	3	2	6	23	3	12	3	10
12	14	15	25	9	13	4	2	10	21	8	19	9	4	7	8	10	25	10	4
1	17	6	15	7	5	7	19	1	20	9	2	14	10	15	16	1	9	9	7
1	20	13	17	5	13	2	9	12	9	3	3	12	22	6	11	3	1	11	21
1	16	6	2	11	6	8	21	7	20	9	3	10	22	11	3	11	12	6	14
4	19	1	6	15	25	7	9	11	7	4	16	12	4	8	3	1	8	8	11
4	14	14	3	6	4	10	14	2	19	14	25	14	6	1	16	10	2	9	23
10	2	7	12	1	7	11	11	10	20	6	11	3	13	4	12	11	11	10	10
3	1	10	15	7	25	7	15	2	13	1	23	14	16	12	1	14	24	9	15
6	10	10	1	8	22	3	20	9	18	5	25	12	10	12	21	10	15	12	3
11	20	11	12	13	4	14	13	2	7	2	19	4	13	8	10	4	19	7	16
6	20	4	2	10	2	7	12	15	25	8	15	5	18	8	11	5	11	15	16
1	3	1	11	9	16	3	9	14	18	14	2	1	23	7	20	14	7	5	20
15	7	8	2	10	20	7	2	1	23	7	7	14	24	4	3	6	23	7	2
4	13	2	18	9	14	10	25	11	19	12	20	14	23	9	1	12	6	9	8
12	19	5	2	1	16	13	13	8	4	11	24	7	10	11	4	15	15	7	18
4	15	15	6	6	5	2	7	4	16	11	17	14	2	11	3	14	23	15	4
4	16	3	23	10	4	12	23	15	4	12	16	14	1	4	22	13	19	5	6
1	14	6	20	10	4	12	7	1	15	13	11	10	13	8	22	7	11	13	23
4	10	1	24	11	8	14	8	11	19	4	14	8	2	7	13	7	1	2	10
8	24	5	3	4	6	13	20	5	10	14	11	8	21	9	9	9	21	3	8
8	6	12	14	13	21	6	2	4	16	15	23	12	1	13	17	3	24	5	7
11	9	11	9	11	9	3	2	2	13	15	24	10	9	7	15	2	8	9	3
8	17	4	7	5	9	12	2	15	24	5	6	2	11	9	21	15	18	8	20
11	23	15	10	12	1	6	15	7	17	8	8	7	4	6	22	3	1	6	9
8	15	4	3	11	14	13	22	12	13	7	21	10	12	2	11	15	7	13	14
15	21	5	4	10	16	15	4	10	13	12	16	15	6	10	16	13	24	2	14
7	21	6	9	3	22	3	1	4	10	1	24	2	14	7	17	8	11	4	19
2	1	3	3	15	4	4	19	6	11	5	25	13	25	11	4	3	23	4	1
7	25	15	10	15	8	5	23	13	21	4	19	4	1	11	8	10	12	12	24
1	22	12	15	3	23	6	14	14	17	1	23	2	11	5	2	3	12	6	12
6	18	13	16	9	4	6	4	13	3	15	21	14	21	5	11	7	9	10	6
6	21	8	20	2	20	8	18	13	1	3	2	14	14	12	17	7	20	13	1
5	7	9	14	12	10	8	18	8	1	11	14	6	12	9	19	8	7	12	6
7	3	2	25	15	24	5	12	9	4	1	15	8	8	11	24	8	18	6	5
15	9	2	20	11	1	4	6	10	21	4	15	7	8	1	17	12	8	11	16
1	22	9	1	11	9	4	18	1	18	5	8	1	15	13	22	8	20	3	17
10	12	2	9	9	13	7	11	2	14	15	3	9	19	13	22	2	16	1	18
15	9	13	9	5	14	14	7	11	2	12	10	14	10	4	10	8	17	13	9
5	15	1	25	6	21	15	25	3	1	14	12	6	6	8	14	5	2	6	3
7	17	2	17	4	3	7	23	11	16	1	6	5	19	1	22	9	24	1	20

Табела 58: Инстанца од 500 пакета

m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V
7	21	5	4	14	13	6	13	10	18	11	1	3	4	5	25	11	17	8	13
6	23	8	20	11	15	1	14	2	8	11	20	12	13	4	1	6	3	1	9
9	1	1	4	7	23	3	17	4	17	13	14	9	5	7	4	7	18	1	15
14	15	11	18	6	9	1	8	5	23	4	4	13	8	13	8	10	13	3	7
2	19	15	12	4	21	15	3	10	23	2	3	1	12	8	16	10	4	7	18
11	9	7	13	8	18	15	2	8	17	7	13	3	7	10	16	12	23	1	23
1	23	14	8	3	23	13	1	3	22	11	12	8	17	11	24	8	19	15	5
2	9	13	17	6	24	9	3	9	10	3	11	6	22	14	10	1	2	2	5
6	18	6	12	11	14	1	13	8	4	3	25	7	18	1	17	11	6	10	13
5	11	7	3	13	6	7	1	7	11	15	3	2	24	12	3	5	8	7	19
2	20	14	16	4	17	3	18	5	21	1	2	5	18	13	4	7	16	6	8
4	2	9	8	4	14	5	6	14	6	6	16	7	9	12	1	2	15	4	1
3	16	6	22	2	7	1	22	14	9	3	22	4	4	12	10	9	24	3	4
7	24	2	4	3	9	3	25	4	11	8	7	4	10	14	12	9	12	14	7
4	7	11	25	15	20	13	9	6	3	8	2	2	25	5	7	14	23	15	19
12	2	3	4	10	18	13	15	3	12	12	11	3	24	7	18	7	9	7	25
2	18	7	12	1	16	8	25	4	20	12	22	5	4	15	2	2	10	13	20
12	19	12	12	10	25	2	18	8	8	1	18	10	13	3	4	15	4	3	13
10	19	13	11	2	10	8	19	6	18	3	4	11	24	5	23	9	9	8	21
6	8	15	12	7	5	3	1	15	20	9	9	4	1	1	2	15	6	11	10
5	14	10	18	1	25	5	8	12	11	4	8	8	5	4	5	1	12	13	3
7	24	5	22	6	7	13	22	15	13	12	13	8	16	8	17	3	13	3	23
1	16	8	8	2	1	2	3	15	23	10	7	11	22	6	2	13	25	6	14
3	21	1	14	12	15	11	25	10	12	7	2	12	5	15	6	1	2	5	13
2	13	3	8	3	4	13	1	14	2	15	2	15	25	12	20	1	23	13	9
7	22	9	18	2	4	8	5	6	6	3	6	8	17	5	11	12	15	5	20
4	20	15	16	13	5	15	19	14	18	8	12	3	1	12	8	1	11	7	16
5	9	3	12	8	9	13	11	11	4	15	22	11	11	4	9	13	15	13	19
15	20	1	3	15	1	11	13	12	18	15	8	9	24	9	18	1	22	7	2
13	6	2	9	12	12	12	14	12	13	9	15	10	25	11	2	12	19	10	6
13	1	4	24	8	25	4	5	2	7	1	8	9	9	7	13	13	7	13	17
11	5	5	4	11	10	14	3	12	23	15	19	3	17	15	18	7	10	14	16
10	13	10	14	13	2	5	18	6	11	11	16	14	22	12	9	14	17	2	24
9	19	15	16	15	13	14	3	7	13	4	9	9	12	15	16	10	23	3	19
7	1	15	3	5	1	8	1	1	19	4	3	4	21	9	23	4	7	12	9
2	23	9	25	9	1	4	5	4	20	4	8	4	8	10	15	7	23	8	12
3	7	9	3	8	1	8	18	11	3	15	8	4	24	11	10	15	24	6	23
10	22	12	11	4	12	10	6	6	22	6	14	15	8	9	22	3	24	15	14
4	14	7	6	3	17	9	24	3	20	3	13	2	4	1	21	14	14	5	6
7	1	1	15	5	10	7	18	8	16	9	16	10	3	5	6	3	18	4	24
11	7	11	12	3	2	14	25	8	18	4	15	15	20	12	2	6	13	13	10
5	8	7	17	2	14	3	14	3	14	2	24	12	2	7	21	5	7	2	8
7	3	15	6	7	21	6	7	3	20	8	2	5	8	1	14	10	5	9	2
3	14	11	17	6	3	12	2	15	13	5	19	1	23	4	24	3	6	2	25
14	15	14	22	13	21	9	23	2	8	14	11	8	3	9	14	4	18	6	14
4	13	8	13	8	21	1	22	10	22	6	24	12	12	4	25	4	12	8	11
11	10	12	10	9	2	14	22	4	20	10	9	14	22	7	18	3	14	9	23
7	23	2	4	10	1	7	15	15	16	8	18	10	5	3	13	15	13	9	20
7	22	12	15	8	19	10	21	6	21	8	17	12	20	15	22	3	23	2	24
11	13	12	15	7	3	1	1	8	22	5	19	12	25	7	9	4	17	1	12

Табела 59: Инстанца од 750 пакета, првих 500

m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V
4	19	3	11	12	18	1	22	14	10	9	1	14	23	13	11	9	24	15	25
9	20	6	21	10	24	6	24	1	17	9	13	9	2	13	7	12	9	12	24
15	24	7	13	12	21	2	21	6	9	14	23	13	8	3	8	4	17	3	16
10	10	10	24	14	15	12	23	3	12	6	4	1	4	7	19	8	13	6	16
11	4	3	10	11	22	12	23	4	4	15	14	5	6	6	1	3	23	2	2
1	5	6	14	4	1	7	25	5	15	4	18	11	20	14	8	1	5	8	6
14	10	1	8	2	5	4	12	11	10	9	18	4	20	6	14	4	18	3	2
7	5	12	7	9	19	2	5	10	6	10	5	8	12	13	15	5	11	15	20
14	17	13	11	9	12	5	23	9	24	8	15	6	10	13	2	13	4	11	3
5	3	11	5	3	19	4	3	8	19	2	3	2	11	12	4	15	17	14	5
1	22	13	1	10	4	14	8	3	17	11	25	12	15	12	21	1	23	13	9
5	13	15	19	11	3	15	18	12	15	13	16	13	8	8	6	12	25	15	13
4	17	11	1	11	18	1	2	9	16	15	22	3	1	7	21	9	20	13	11
5	11	9	2	2	10	8	19	8	1	14	23	12	17	12	13	7	1	8	12
11	11	10	9	5	17	15	21	10	16	10	16	4	16	8	4	13	24	5	9
8	14	13	19	3	16	5	22	5	13	7	21	12	16	15	8	10	23	6	2
10	11	15	11	8	24	15	9	1	8	13	25	4	11	1	5	1	21	8	21
4	12	4	17	1	24	12	16	2	16	14	6	10	7	2	25	3	8	4	6
5	8	4	1	14	17	6	18	7	4	8	4	5	2	7	3	12	12	13	16
13	8	13	7	4	16	9	2	2	20	15	13	9	25	13	16	2	3	3	24
9	19	8	22	13	13	7	4	4	15	10	24	15	15	14	23	10	21	10	25
11	3	1	7	15	17	6	18	12	22	4	11	13	8	10	7	6	8	2	10
4	14	14	8	2	13	9	1	9	25	4	18	10	8	12	23	14	16	5	9
3	25	6	10	5	9	7	11	1	19	3	2	2	23	2	20	15	24	12	8
10	12	6	8	4	1	6	22	7	3	10	13	2	14	10	21	2	21	13	13

Табела 60: Инстанца од 750 пакета, пакети $501 - 750$

m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V
3	4	15	22	14	10	4	21	5	17	1	15	1	10	14	11	6	16	4	17
4	21	15	11	3	5	3	16	11	16	2	20	6	13	1	9	3	13	9	22
9	22	6	6	4	23	4	25	12	20	5	24	12	15	11	16	6	11	5	11
6	15	11	11	2	17	3	23	10	8	1	8	10	2	5	18	14	22	8	22
3	3	9	23	13	17	5	17	3	10	1	13	14	20	11	5	11	3	5	9
13	12	13	3	11	24	9	2	1	8	11	8	7	24	4	19	13	23	3	12
12	21	11	15	5	8	7	2	13	6	6	10	10	10	4	2	15	3	13	8
4	17	8	21	8	10	8	1	15	14	7	20	9	21	9	11	6	25	14	18
9	5	9	18	14	5	1	21	3	25	5	12	13	10	5	24	8	4	11	3
3	1	12	12	11	9	6	3	4	5	12	13	9	16	7	1	7	20	3	10
12	14	3	18	12	9	14	14	7	5	12	18	12	8	7	15	10	17	11	5
1	8	2	8	14	9	9	9	12	9	7	23	6	6	4	24	4	13	10	10
13	7	9	15	12	18	10	10	9	2	5	18	15	22	1	2	4	13	12	3
5	7	7	21	7	5	8	15	5	23	9	12	15	17	10	15	7	15	1	1
12	23	15	15	4	6	8	15	3	24	6	20	8	11	14	13	3	14	12	3
5	6	3	8	7	24	6	2	4	20	3	19	12	25	15	9	15	9	2	11
6	12	13	21	12	21	15	19	10	1	4	23	14	1	7	5	10	8	2	25
11	17	6	11	5	20	1	4	15	17	2	17	10	15	10	20	11	10	8	5
8	15	5	8	14	12	1	2	10	7	10	2	2	17	13	2	12	12	8	14
15	13	5	22	8	6	8	21	3	10	5	1	1	23	13	12	13	1	14	13
1	3	4	3	1	9	13	2	7	6	5	10	7	13	6	22	9	16	1	19
3	20	5	12	2	20	1	23	8	10	3	11	2	13	7	25	13	12	8	19
11	2	3	2	13	22	15	23	12	20	7	10	4	6	5	1	4	22	11	2
1	10	13	13	1	22	14	3	14	13	12	14	3	23	4	23	10	7	4	2
12	8	10	13	10	24	10	10	4	1	12	13	12	19	10	9	3	24	3	5
11	18	1	20	7	24	7	18	3	4	1	3	2	8	12	9	14	7	15	14
11	11	12	21	15	10	4	7	5	25	6	5	10	22	3	14	10	7	15	22
5	15	6	4	8	24	7	6	3	12	9	9	1	10	10	6	13	10	2	6
10	3	5	22	12	6	1	17	14	13	15	7	11	24	4	9	10	16	13	10
8	12	8	25	13	2	6	13	11	22	3	16	8	7	3	23	10	3	9	2
13	20	2	5	1	14	15	1	11	12	5	12	5	5	6	23	4	10	7	20
5	23	7	19	2	14	15	11	9	25	7	9	3	1	2	3	3	16	1	16
3	14	5	10	1	21	14	2	12	8	1	16	4	25	8	11	12	19	4	11
7	23	9	12	3	20	3	1	13	19	5	9	7	9	2	3	14	25	9	14
7	2	10	23	12	19	10	7	1	22	4	13	8	23	14	23	8	8	7	22
6	20	3	14	12	10	4	1	11	8	12	4	3	23	10	1	4	7	6	7
10	16	2	14	5	25	2	24	3	14	10	12	8	17	15	17	11	2	3	20
2	11	4	9	5	21	11	23	14	2	15	23	11	2	13	17	3	10	2	2
12	1	1	19	4	2	14	20	13	12	11	15	3	15	5	1	14	22	1	1
12	7	12	25	10	2	14	19	2	8	15	11	15	14	6	2	5	9	7	16
12	21	1	22	15	10	11	10	14	3	11	6	6	13	4	15	8	17	13	14
2	24	5	16	14	22	4	23	5	19	3	15	5	25	11	15	14	24	13	9
3	2	7	8	2	19	5	11	4	21	8	21	1	8	9	17	8	11	4	22
8	22	5	9	13	11	3	15	1	17	1	8	14	12	14	2	7	14	14	19
4	25	13	10	10	6	3	12	13	2	9	12	6	19	13	13	7	12	7	18
6	25	2	16	14	12	12	11	9	9	7	20	6	11	2	2	1	22	10	16
12	24	11	20	2	3	5	5	3	7	15	16	8	21	4	4	13	9	8	5
11	2	11	13	3	9	6	8	12	24	1	16	1	2	6	4	4	14	7	7
7	3	4	3	13	23	15	18	9	17	10	25	6	13	11	3	15	17	4	17
6	7	5	19	8	9	1	25	2	8	10	4	14	21	4	2	11	13	1	15

Табела 61: Инстанца од 1000 пакета, првих 500

m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V	m	V
6	10	13	5	9	8	10	2	1	18	14	19	12	21	9	3	10	17	2	24
4	6	3	10	9	8	9	6	4	17	4	2	10	9	5	17	3	1	9	14
6	1	6	1	14	3	15	5	6	15	2	25	2	16	10	1	10	21	15	10
11	17	10	8	1	18	7	19	6	13	12	4	2	22	14	15	5	18	1	15
7	23	11	3	10	17	11	25	14	20	2	16	13	8	9	11	4	10	7	19
6	18	9	16	7	17	12	13	14	22	11	19	5	23	11	11	13	24	6	6
13	21	9	8	6	6	9	17	13	5	9	8	15	15	10	16	2	25	3	25
8	12	5	21	4	15	9	7	14	5	14	21	6	1	4	22	9	23	9	21
7	13	10	23	15	15	12	15	15	7	11	5	12	14	3	21	14	11	10	10
14	23	3	20	8	22	4	21	13	20	2	9	12	12	10	12	14	23	7	8
5	8	12	6	11	15	9	5	11	25	13	2	8	3	14	6	2	14	8	20
5	16	12	11	9	4	11	23	11	9	6	24	13	16	2	2	7	15	15	5
14	19	7	21	8	12	11	15	1	13	10	19	8	6	7	3	9	9	14	18
6	6	3	8	15	19	2	1	6	1	1	10	1	21	13	23	11	11	13	10
11	9	5	17	15	23	9	23	11	18	9	14	12	14	1	13	14	13	2	13
13	9	13	10	6	18	7	6	12	25	2	2	8	21	11	19	15	19	3	23
15	14	15	3	10	21	14	5	15	15	10	25	7	11	7	4	13	17	3	12
12	16	12	24	11	22	2	25	10	10	11	22	11	22	3	15	15	16	3	22
11	19	2	24	1	25	1	7	11	12	12	20	6	23	8	25	4	12	15	9
4	1	14	6	3	21	7	8	9	24	5	5	1	23	12	2	9	18	11	3
3	15	11	8	11	10	11	19	5	20	7	25	12	16	13	24	7	7	10	23
15	14	3	5	13	2	14	6	2	14	15	20	1	4	6	2	7	10	15	10
11	25	5	17	11	13	5	6	5	24	2	1	13	18	10	22	7	11	8	8
14	5	11	11	5	20	13	3	6	22	8	2	9	7	1	6	8	4	13	7
7	15	1	5	13	15	5	13	11	16	13	5	15	4	9	11	13	9	3	25
12	10	10	6	15	2	8	17	10	17	15	8	5	21	12	2	15	20	4	24
12	23	9	8	8	12	15	17	8	21	6	13	9	4	14	5	5	18	4	2
3	7	10	7	6	16	10	21	3	20	4	22	10	18	15	14	4	1	4	25
4	6	8	25	7	25	13	18	1	19	10	4	6	9	10	19	11	3	3	25
3	16	2	18	14	7	6	20	2	7	4	11	4	2	7	17	11	4	10	12
12	20	6	16	11	14	15	20	13	5	3	8	5	16	15	23	6	25	15	2
1	6	4	15	9	22	10	1	6	13	2	5	10	9	6	3	15	21	2	19
13	16	9	19	4	4	2	12	1	17	6	10	7	25	3	1	11	18	8	1
6	9	15	2	7	13	4	8	5	16	12	2	5	2	14	11	9	19	2	7
9	10	1	15	9	20	1	21	3	1	4	8	9	9	10	25	10	7	15	2
14	12	3	22	3	10	13	24	1	21	1	21	3	13	12	5	12	7	6	15
5	8	5	11	8	5	14	21	5	4	15	24	2	7	8	21	11	1	8	25
12	22	2	11	14	24	8	18	5	21	14	7	10	1	6	2	14	24	5	15
6	12	8	18	5	23	12	20	14	6	10	25	6	20	4	15	14	20	9	20
12	17	1	21	6	2	10	19	4	21	4	19	13	11	7	12	9	1	13	1
2	7	8	10	1	24	4	9	11	15	13	8	14	2	15	25	9	3	8	9
11	7	12	3	1	9	6	10	12	25	10	14	5	21	3	20	6	24	1	3
14	21	1	14	4	15	7	8	11	1	9	21	2	12	15	9	2	20	9	9
10	7	4	10	1	9	15	22	15	13	12	6	13	11	12	9	2	5	13	12
10	24	11	15	1	25	13	14	15	9	5	21	10	25	5	14	1	19	5	20
11	10	5	3	8	13	1	19	6	14	8	21	12	21	13	21	11	3	10	3
5	7	4	4	13	25	3	17	1	16	1	22	14	1	15	12	10	8	6	23
9	21	8	11	8	11	12	18	3	17	13	23	14	19	11	2	13	21	6	21
10	10	10	18	4	1	7	21	2	24	15	16	12	5	6	11	1	3	14	2
10	12	2	8	14	1	13	15	11	7	3	2	8	22	2	17	9	1	12	16

Табела 62: Инстанца од 1000 пакета, других 500

Литература

[1] Bernard T Han, George Diehr, and Jack S Cook. Multiple-type, two-dimensional bin packing problems: Applications and algorithms. *Annals of Operations Research*, 50(1):239–261, 1994.

[2] Michaël Gabay and Sofia Zaourar. Vector bin packing with heterogeneous bins: application to the machine reassignment problem. *Annals of Operations Research*, 242(1):161–194, 2016.

[3] Alberto Caprara and Paolo Toth. Lower bounds and algorithms for the 2-dimensional vector packing problem. *Discrete Applied Mathematics*, 111(3):231–262, 2001.

[4] Filipe Brandao and João Pedro Pedroso. Bin packing and related problems: general arc-flow formulation with graph compression. *Computers & Operations Research*, 69:56–67, 2016.

[5] Đorđe Stakić, Ana Anokić, and Raka Jovanovic. Comparison of different grasp algorithms for the heterogeneous vector bin packing problem. In *2019 China-Qatar International Workshop on Artificial Intelligence and Applications to Intelligent Manufacturing (AIAIM)*, pages 63–70. IEEE, 2019.

[6] GHG Protocol Mobile Guide. Calculating CO2 Emissions from Mobile Sources. Technical report, Greenhouse Gas Protocol, 2005.

[7] Teodor Gabriel Crainic and Kap Hwan Kim. Intermodal transportation. *Handbooks in operations research and management science*, 14:467–537, 2007.

[8] Tolga Bektas and Teodor Crainic. *A brief overview of intermodal transportation.* CIRRELT, 2007.

[9] Michael Berliner Pedersen, Oli BG Madsen, and Otto Anker Nielsen. *Optimization models and solution methods for intermodal transportation.* PhD thesis, Technical University of DenmarkDanmarks Tekniske Universitet, Department of TransportInstitut for Transport, Traffic ModellingTrafikmodeller, 2005.

[10] Manish Verma, Vedat Verter, and Nicolas Zufferey. A bi-objective model for planning and managing rail-truck intermodal transportation of hazardous materials. *Transportation research part E: logistics and transportation review*, 48(1):132–149, 2012.

[11] Nam Kim, Milan Janic, and Bert Van Wee. Trade-off between carbon dioxide emissions and logistics costs based on multiobjective optimization. *Transportation Research Record: Journal of the Transportation Research Board*, (2139):107–116, 2009.

[12] Yann Bouchery and Jan Fransoo. Cost, carbon emissions and modal shift in intermodal network design decisions. *International Journal of Production Economics*, 164:388–399, 2015.

[13] Jasmine Siu Lee Lam and Yimiao Gu. Port hinterland intermodal container flow optimisation with green concerns: a literature review and research agenda. *International Journal of Shipping and Transport Logistics*, 5(3):257–281, 2013.

[14] M SteadieSeifi, Nico P Dellaert, W Nuijten, Tom Van Woensel, and R Raoufi. Multimodal freight transportation planning: A literature review. *European journal of operational research*, 233(1):1–15, 2014.

[15] Stanley E Griffis, John E Bell, and David J Closs. Metaheuristics in logistics and supply chain management. *Journal of Business Logistics*, 33(2):90–106, 2012.

[16] Y Cardona-Valdés, A Álvarez, and J Pacheco. Metaheuristic procedure for a bi-objective supply chain design problem with uncertainty. *Transportation Research Part B: Methodological*, 60:66–84, 2014.

[17] Ali Mohtashami, Madjid Tavana, Francisco J Santos-Arteaga, and Ali Fallahian-Najafabadi. A novel multi-objective meta-heuristic model for solving cross-docking scheduling problems. *Applied Soft Computing*, 31:30–47, 2015.

[18] Stefan Ropke. Heuristic and exact algorithms for vehicle routing problems. *Unpublished PhD thesis, Computer Science Department, University of Copenhagen*, 2005.

[19] Radoslav Rajković, Nenad Zrnić, Djordje Stakić, and Borut Mahnič. The costs of container transport flow between Far East and Serbia using different liner shipping services. *Logistics & Sustainable Transport*, 6(1):34–40, 2015.

[20] Radoslav Rajković, Nenad Zrnić, and Djordje Stakić. Different approaches for minimizing transport costs in intermodal networks. In *13th International Logistics and Supply Chain Congress - Proceedings, Izmir, Turkey, 22nd-23rd October 2015*, pages 160–167, 2015.

[21] Radoslav Rajković, Nenad Zrnić, and Djordje Stakić. Contribution to optimal container flow routing between Far East and Serbia through selected Adriatic ports. In *Proceedings of the 5th International Conference Transport and Logistics TIL 2014, Ni, Serbia, 05-06. June 2014*, pages 87–91, 2014.

[22] Radoslav Rajković, Nenad Zrnić, Djordje Stakić, Aleksandar Sedmak, and Snežana Kirin. Environmental protection in intermodal networks by minimizing CO2 emission. In *Proceedings of TEAM 2015, 7th International Scientific and Expert Conference of the International TEAM Society 1516th October 2015, Belgrade, Serbia*, pages 274–278, 2015.

[23] Djordje Stakić, Radoslav Rajković, Dušan Tošić, and Nenad Zrnić. Evaluation of pareto optimal solutions in intermodal networks. In *42nd International Symposium on Operations Research - Proceedings, SYM-OP-IS 2015, September 15-18, 2015, Srebrno jezero, Serbia*, pages 319–322, 2015.

[24] Radoslav Rajković, Nenad Zrnić, Branislav Dragović, and Djordje Stakić. Multi-criteria decision making methods in container transport. In *XXI International conference on material handling constructions and logistics MHCL 15, September 23-25, 2015, Vienna, Austria*, pages 289–294, 2015.

[25] Radoslav Rajković, Nenad Zrnić, and Djordje Stakić. Application of mathematical model for container transport flow of goods: From Far East to Serbia. In *Proceedings of 12th International Conference on Industrial Logistics ICIL 2014, Bol on island Brac, Croatia, 11-13, June 2014*, pages 159–166, 2014.

[26] Radoslav Rajković, Nenad Zrnić, and Djordje Stakić. Application of mathematical model for container transport flow of goods: from Far East to Serbia. *Tehnički vjesnik*, 23(6):1739–1746, 2016.

[27] Radoslav Rajković, Nenad Zrnić, Olja Čokorilo, Snežana Rajković, and Djordje Stakić. Multi-objective container transport optimization on intermodal networks based on mathematical model. In *Proceeding of the International Conference on Trafic and Transport Engineering ICTTE 2014, Beograd, Serbia, 27-28. November 2014*, pages 26–36, 2014.

[28] Радослав Рајковић, Ђорђе Стакић. Примена математичког модела: вишекритеријумска оптимизација у контејнерском транспорту. In *Зборник радова - V симпозијум Математика и примене, 17. и 18. октобар 2014, Београд, Србија*, pages 37–47, 2015.

[29] ISO 668:1995 Series 1 freight containers Classification, dimensions and ratings AMENDMENT 2: 45' containers. Technical report, International Organization for Standardization (ISO), 2005.

[30] Radoslav Rajković, Nenad Zrnić, Djordje Stakić, Aleksandar Sedmak, and Snežana Kirin. An approach to determine optimal number of containers for cargo stacking in function of transportation cost. In *Proceedings - 6th International Symposium on Industrial Engineering - SIE 2015, 24th-25th September 2015, Belgrade, Serbia*, pages 300–303, 2015.

[31] Alan McKinnon and Maja Piecyk. Measuring and managing CO2 emissions of European Chemical Transport. Technical report, Edinburgh: European Chemical Industry Council, 2010.

[32] Hans Kellerer, Ulrich Pferschy, and David Pisinger. *Knapsack problems*. Springer, 2004.

[33] David Pisinger. Algorithms for knapsack problems. 1995.

[34] Silvano Martello and Paolo Toth. *Knapsack problems: algorithms and computer implementations*. John Wiley & Sons, Inc., 1990.

[35] Mirko Vujošević, Milan Stanojević, and Nenad Mladenović. *Metode optimizacije: mrežni, lokacijski i višekriterijumski modeli*. Društvo operacionih istraživača Jugoslavije, 1996.

[36] Xin-She Yang (Auth.). *Nature-Inspired Optimization Algorithms*. Elsevier Insights. Elsevier, 1 edition, 2014.

[37] Milan Stanojević. *Višekriterijumski pristup rešavanju Štajnerovog problema na grafu*. Fakultet organizacionih nauka Univerziteta u Beogradu, doktorska disertacija, 2005.

[38] Manuel Iori, Juan-José Salazar-González, and Daniele Vigo. An exact approach for the vehicle routing problem with two-dimensional loading constraints. *Transportation Science*, 41(2):253–264, 2007.

[39] Manuel Iori and Silvano Martello. Routing problems with loading constraints. *Top*, 18(1):4–27, 2010.

[40] Andreas Bortfeldt and Gerhard Wäscher. Constraints in container loading–a state-of-the-art review. *European Journal of Operational Research*, 229(1):1–20, 2013.

[41] Arnaud Fréville. The multidimensional 0–1 knapsack problem: An overview. *European Journal of Operational Research*, 155(1):1–21, 2004.

[42] Radoslav Rajković, Nenad Zrnić, Sanja Bojić and Djordje Stakić. Role of cargo weight and volume: Minimizing costs and CO2 emissions in container transport. In *Commercial Transport, Proceedings of the 2nd Interdiciplinary Conference on Production Logistics and Traffic 2015*, Lecture Notes in Logistics, pages 159–173. Springer International Publishing, 2016.

[43] Gerhard Wäscher, Heike Haußner, and Holger Schumann. An improved typology of cutting and packing problems. *European journal of operational research*, 183(3):1109–1130, 2007.

[44] Maxence Delorme, Manuel Iori, and Silvano Martello. Bpplib: a library for bin packing and cutting stock problems. *Optimization Letters*, 12(2):235–250, 2018.

[45] Maxence Delorme, Manuel Iori, and Silvano Martello. Bin packing and cutting stock problems: Mathematical models and exact algorithms. *European Journal of Operational Research*, 255(1):1–20, 2016.

[46] Edward G Coffman Jr, János Csirik, Gábor Galambos, Silvano Martello, and Daniele Vigo. Bin packing approximation algorithms: survey and classification. *Handbook of combinatorial optimization*, pages 455–531, 2013.

[47] Michaël Gabay and Sofia Zaourar. Variable size vector bin packing heuristics-application to the machine reassignment problem. 2013.

[48] Michael R Garey and David S Johnson. Computers and intractability: a guide to np-completeness, 1979.

[49] Djordje Stakić, Miodrag Živković, and Ana Anokić. Instances for the two-dimensional heterogeneous vector bin packing problem, August 2021.

[50] S. Feo and M.G.C. Resende. A probabilistic heuristic for a computationally diffcult set covering problem. *Oper. Res. Lett.*, 8:67–71, 1989.

[51] S. Feo and M.G.C. Resende. Greedy randomized adaptive search procedures. *J. Global Optim.*, 6:109–133, 1995.

[52] M.G.C. Resende and C.C. Ribeiro. Greedy randomized adaptive search procedures: Advances, hybridizations, and applications. In M. Gendreau and Potvin J.Y., editors, *Handbook of metaheuristics*, pages 283–319. Springer, 2010.

[53] P. Festa and M.G.C. Resende. An annotated bibliography of GRASP–part I: Algorithms. *Int. T. Oper. Res.*, 16(1):1–24, 2009.

[54] M.G.C. Resende and C.C. Ribeiro. Grasp: Greedy randomized adaptive search procedures. In E.K. Burke and G. Kendall, editors, *Search Methodologies - Introductory tutorials in optimization and decision support systems*, pages 287–312. Springer, 2014.

[55] P. Festa and M. G.C. Resende. An annotated bibliography of GRASP–part II: Applications. *Int. T. Oper. Res.*, 16(2):131–172, 2009.

[56] Xavier Delorme, Xavier Gandibleux, and Joaquin Rodriguez. GRASP for set packing problems. *European Journal of Operational Research*, 153(3):564–580, 2004.

[57] Marcelo Prais and Celso C Ribeiro. Reactive grasp: An application to a matrix decomposition problem in tdma traffic assignment. *INFORMS Journal on Computing*, 12(3):164–176, 2000.

[58] N. Mladenović and P. Hansen. Variable neighborhood search. *Comput. Oper. Res.*, 24(11):1097–1100, 1997.

[59] P. Hansen and N. Mladenović. Variable neighborhood search: principles and applications. *Eur. J. Oper. Res.*, 130(3):449–467, 2001.

[60] P. Hansen, N. Mladenović, and J. A.M. Pérez. Variable neighbourhood search: methods and applications. *Ann. Oper. Res.*, 175(1):367–407, 2010.

[61] P. Hansen and N. Mladenović. Variable neighborhood search. In E. K. Burke and R. D. Graham, editors, *Search methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, pages 313–337. Springer-Verlag, New York, 2014.

[62] Djordje Stakić, Miodrag Živković, Ana Anokić, and Radoslav Rajković. Rešavanje problema pakovanja paketa u kontejnere uz ograničenja mase i zapremine metodom vns. In *45nd International Symposium on Operations Research - Proceedings, SYM-OP-IS 2018, September 16-18, 2018, Zlatibor, Serbia*, pages 112–117, 2018.

[63] Djordje Stakić, Miodrag Živković, and Ana Anokić. A reduced variable neighborhood search approach to the heterogeneous vector bin packing problem. *Information Technology and Control*, 50(4):808–826, 2021.

[64] Katrin Heßler, Timo Gschwind, and Stefan Irnich. Stabilized branch-and-price algorithms for vector packing problems. *European Journal of Operational Research*, 271(2):401–419, 2018.

Биографија аутора

Ђорђе Стакић је рођен 3. марта 1982. године у Крупњу, где је завршио основну школу као ученик генерације. Током школовања освајао је награде на републичком и савезном такмичењу из математике и републичком такмичењу из хемије. После тога је завршио Математичку гимназију у Београду, затим Математички факултет Универзитета у Београду где је 17.1.2007. дипломирао на смеру Рачунарство и информатика са просечном оценом 9,72. Исте 2007. године уписао је докторске студије на Математичком факултету, а од 2009. је студент докторских студија информатике по новом студијском програму. У периоду од 2007. до 2009. два пута је биран у звање сарадника у настави на Катедри за рачунарство и информатику Математичког факултета. Током тог периода држао је вежбе из предмета: Програмирање 1, Програмирање 2, Преводиоци и интерпретатори и Превођење програмских језика. После одслужења војне обавезе, током школске 2009/10 године, од 2010. године био је запослен у Рачунарској лабораторији Математичког факултета. Био је члан организационог одбора симпозијума Математика и примене на Математичком факултету од 2011. до 2021. године, као и члан организационог одбора конференције SYMOPIS 2018. и 2022. Током две школске године 2012/13 и 2013/14 држао је факултативни курс Елементарна математика за студенте прве године Математичког факултета. Школске 2008/09 године хонорарно је био ангажован као професор у Математичкој гимназији у Београду. Иницијатор је идеје да се Википедија користи као платформа за студентске семинарске радове. На тај начин је остварио сарадњу са више факултета и средњих школа. Аутор је и реализатор семинара за стручно усавршавање наставника, као и семинара за обуку библиотекара (Вики-библиотекар). Од 2014. године се у научном смислу највише бави оптимизацијама које имају примене у транспорту. Из тога је проистекло више радова на међународним конференцијама и у научним часописима. Од 2018. године је у звању асистента на Катедри за статистику и математику Економског факултета Универзитета у Београду где је биран за предмете: Пословна информатика, Математика и Финансијска и актуарска математика. Осим њих држао је вежбе из предмета: Објектно оријентисано програмирање, Информациони системи и пословна аналитика, Електронско пословање, ЕРП софтвер и Нове информационе технологије.

Прилог 1.

Изјава о ауторству

Потписани-а Ђорђе Стакић

број уписа 2050/2009

Изјављујем

да је докторска дисертација под насловом

Математички модели и различити начини вишекритеријумске оптимизације у интермодалном транспорту

- резултат сопственог истраживачког рада,
- да предложена дисертација у целини ни у деловима није била предложена за добијање било које дипломе према студијским програмима других високошколских установа,
- да су резултати коректно наведени и
- да нисам кршио/ла ауторска права и користио интелектуалну својину других лица.

Потпис докторанда

У Београду, 23.5.2022.

Ђорђе Стакић

Прилог 2.

Изјава о истоветности штампане и електронске верзије докторског рада

Име и презиме аутора Ђорђе Стакић

Број уписа 2050/2009

Студијски програм Информатика

Наслов рада Математички модели и различити начини вишекритеријумске оптимизације у интермодалном транспорту

Ментор проф. др Миодраг Живковић

Потписани Ђорђе Стакић

изјављујем да је штампана верзија мог докторског рада истоветна електронској верзији коју сам предао/ла за објављивање на порталу **Дигиталног репозиторијума Универзитета у Београду.**

Дозвољавам да се објаве моји лични подаци везани за добијање академског звања доктора наука, као што су име и презиме, година и место рођења и датум одбране рада.

Ови лични подаци могу се објавити на мрежним страницама дигиталне библиотеке, у електронском каталогу и у публикацијама Универзитета у Београду.

Потпис докторанда

У Београду, 23.5.2022.

Ђорђе Стакић

Прилог 3.

Изјава о коришћењу

Овлашћујем Универзитетску библиотеку „Светозар Марковић“ да у Дигитални репозиторијум Универзитета у Београду унесе моју докторску дисертацију под насловом:

Математички модели и различити начини вишекритеријумске оптимизације у интермодалном транспорту

која је моје ауторско дело.

Дисертацију са свим прилозима предао/ла сам у електронском формату погодном за трајно архивирање.

Моју докторску дисертацију похрањену у Дигитални репозиторијум Универзитета у Београду могу да користе сви који поштују одредбе садржане у одабраном типу лиценце Креативне заједнице (Creative Commons) за коју сам се одлучио/ла.

1. Ауторство
2. Ауторство - некомерцијално
3. Ауторство – некомерцијално – без прераде
4. Ауторство – некомерцијално – делити под истим условима
5. Ауторство – без прераде
6. Ауторство – делити под истим условима

(Молимо да заокружите само једну од шест понуђених лиценци, кратак опис лиценци дат је на полеђини листа).

Потпис докторанда

У Београду, 23.5.2022.

Жорже Стојић

1. Ауторство - Дозвољавате умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце, чак и у комерцијалне сврхе. Ово је најслободнија од свих лиценци.

2. Ауторство – некомерцијално. Дозвољавате умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце. Ова лиценца не дозвољава комерцијалну употребу дела.

3. Ауторство - некомерцијално – без прераде. Дозвољавате умножавање, дистрибуцију и јавно саопштавање дела, без промена, преобликовања или употребе дела у свом делу, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце. Ова лиценца не дозвољава комерцијалну употребу дела. У односу на све остале лиценце, овом лиценцом се ограничава највећи обим права коришћења дела.

4. Ауторство - некомерцијално – делити под истим условима. Дозвољавате умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце и ако се прерада дистрибуира под истом или сличном лиценцом. Ова лиценца не дозвољава комерцијалну употребу дела и прерада.

5. Ауторство – без прераде. Дозвољавате умножавање, дистрибуцију и јавно саопштавање дела, без промена, преобликовања или употребе дела у свом делу, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце. Ова лиценца дозвољава комерцијалну употребу дела.

6. Ауторство - делити под истим условима. Дозвољавате умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце и ако се прерада дистрибуира под истом или сличном лиценцом. Ова лиценца дозвољава комерцијалну употребу дела и прерада. Слична је софтверским лиценцама, односно лиценцама отвореног кода.