UNICAMP

RODRIGO MORETTI BRANCHINI

"FLEET DEPLOYMENT OPTIMIZATION IN LINER SHIPPING"

"OTIMIZAÇÃO DO DIMENSIONAMENTO E ROTEAMENTO DE NAVIOS DE LINHA REGULAR COM VIAGENS FRETADAS"

CAMPINAS
2013

UNICAMP

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

RODRIGO MORETTI BRANCHINI

"FLEET DEPLOYMENT OPTIMIZATION IN LINER SHIPPING"

Orientador: Prof. Dr. Vinícius Amaral Armentano

"*OTIMIZAÇÃO DO DIMENSIONAMENTO E ROTEAMENTO DE NAVIOS DE LINHA REGULAR COM VIAGENS FRETADAS*"

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas para obtenção do título de Doutor em Engenharia Elétrica, na área de Automação.

Doctorate thesis presented to the Postgraduate Programme of Electrical Engineering of the School of Electrical Engineering of the State University of Campinas to obtain the Ph.D. degree in Electrical Engineering, in the field of Automation.

ESTE EXEMPLAR CORRESPONDE À VERSÃO FINAL DA TESE
DEFENDIDA PELO ALUNO RODRIGO MORETTI BRANCHINI,
E ORIENTADA PELO PROF. DR. VINÍCIUS AMARAL ARMENTANO.

CAMPINAS
2013

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Elizangela Aparecida dos Santos Souza - CRB 8/8098

	Branchini, Rodrigo Moretti, 1975-
B732o	Otimização do dimensionamento e roteamento de navios de linha regular com viagens fretadas / Rodrigo Moretti Branchini. – Campinas, SP : [s.n.], 2013.
	Orientador: Vinícius Amaral Armentano.
	Tese (doutorado) – Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.
	1. Logística. 2. Transporte marítimo. 3. Programação inteira. 4. Metaheurística. 5. Busca tabu. I. Armentano, Vinícius Amaral,1950-. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: Fleet deployment optimization in liner shipping
Palavras-chave em inglês:
Logistics
Maritime transportation
Integer programming
Metaheuristic
Tabu search
Área de concentração: Automação
Titulação: Doutor em Engenharia Elétrica
Banca examinadora:
Vinícius Amaral Armentano [Orientador]
Reinaldo Morábito Neto
Franklina Maria Bragion de Toledo
Flavio Keidi Miyazawa
Paulo Augusto Valente Ferreira
Data de defesa: 21-06-2013
Programa de Pós-Graduação: Engenharia Elétrica

COMISSÃO JULGADORA - TESE DE DOUTORADO

Candidato: Rodrigo Moretti Branchini

Data da Defesa: 21 de junho de 2013

Título da Tese: "Otimização do Dimensionamento e Roteamento de Navios de Linha Regular com Viagens Fretadas"

Prof. Dr. Vinícius Amaral Armentano (Presidente): Vinícius A. Armentano

Prof. Dr. Reinaldo Morábito Neto: ______________________

Profa. Dra. Franklina Maria Bragion de Toledo: ______________________

Prof. Dr. Flavio Keidi Miyazawa: ______________________

Prof. Dr. Paulo Augusto Valente Ferreira: ______________________

To my beloved wife Josi

ACKNOWLEDGEMENTS

I would like to thank everyone who contributed to the development of this work, especially my supervisor Professor Vinícius Amaral Armentano, who has guided and helped me during the almost 10 years of my academic career. I owe him many thanks for all his patience, dedication and friendship.

I would like also to thank Ana Armentano, whose help with the review of the English writing is much appreciated. I also appreciated the laughs that the three of us had together in Berlin.

I gratefully acknowledge the support of Kjetil Fagerholt and Lars Magnus Hvattum, who contributed with an initial suggestion of reference papers and also with the distance matrix data that were used on the set of benchmark problems.

I appreciate the comments on this thesis from the PhD examination team: Reinaldo Morabito, Franklina Toledo, Flávio Miyazawa and Paulo Valente. Thank you for taking the time to read this work and for sharing your experience with me.

I am thankful to all Professors of FEEC who have shared their knowledge with me during the PhD classes. The Professors were rigorous and the classes required lots of hard work, but there is no other better way to learn.

Thanks go out to Zilda, Janete and Suzete of the technical support of DENSIS. They readily helped me to overcome those annoying hardware and software bugs that emerge from time to time.

I thank all fellow PhD students that I have had the pleasure to meet during the years at Unicamp. The discussions we had at classes and DENSIS contributed to the completion of this work.

I am also grateful for the initial financial support of CNPq.

I am very thankful for all the support that my family, mother and brother have given me during all these years. Thanks for always being caring and by my side.

Finally, I warmly thank my wife, Josi, for the love and backing throughout this venture. I am extremely grateful for her kindness, encouragement and for always being a safe harbor.

"If you want to build a ship, don't drum up the men to gather wood, divide the work and give orders. Instead, teach them to yearn for the vast and endless sea."

(Antoine de Saint-Exupery)

RESUMO

Este trabalho aborda um problema de planejamento tático em empresas de transporte marítimo de carga que coletam e entregam as demandas contratadas por seus clientes. As viagens associadas a estas demandas são obrigatórias, mas a empresa pode também atender a demandas *spot* associadas com viagens opcionais para aumentar seu lucro durante um horizonte de tempo de médio prazo. O problema de otimização é formulado como um modelo de programação inteira mista que é definido em um grafo orientado em que nós representam viagens obrigatórias e opcionais. As decisões do modelo são determinar o número e tipo de navios que compõem a frota, designar um navio a um conjunto de viagens obrigatórias e opcionais, definir as rotas de cada navio e estipular os tempos de início de atendimento nos portos para cada viagem. Um algoritmo de busca tabu com uma lista de candidatos e um conjunto de soluções de elite é proposto para resolver instâncias do problema. Os resultados computacionais da busca tabu são comparados com as soluções ótimas e sub-ótimas encontradas pelo CPLEX para o modelo de programação inteira mista.

Palavras-chave: logística, transporte marítimo, programação inteira, metaheurística, busca tabu.

ABSTRACT

We address a tactical planning problem faced by many liner shipping companies that have committed contractual voyages while trying to serve optional spot voyages to increase its revenue over the medium-term horizon. The optimization problem is formulated as a mixed integer programming model that is defined on a directed graph whose nodes represent contractual and spot voyages. The decisions include the number and type of vessels deployed the assignment of vessels to contractual and spot voyages and the determination of vessel routes and schedules in order to maximize the profit. A tabu search algorithm with a candidate list and a pool of elite and diverse solutions is proposed in order to solve a set of benchmark instances of the problem. The results obtained by tabu search are compared to optimal and suboptimal solutions yielded by the CPLEX solver to the mixed integer programming formulation of the problem.

Keywords: logistics, maritime transportation, integer programming, metaheuristic, tabu search.

LIST OF FIGURES

Introduction:

Paper A - Fleet deployment optimization in liner shipping with spot voyages:

Paper B - Tabu search for fleet deployment and routing in liner shipping:

LIST OF TABLES

Introduction:

Paper A - Fleet deployment optimization in liner shipping with spot voyages:

Paper B - Tabu search for fleet deployment and routing in liner shipping:

LIST OF SYMBOLS

Problem parameters:

ACV_{vi} = 1 if ship $v \in V$ may be assigned to contractual voyage $i \in CV$, 0 otherwise.

ASV_{vr} = 1 if ship $v \in V$ may be assigned to spot voyage $r \in SV$, 0 otherwise.

BCO_v – minimum time (in hours) to charter ship $v \in VCO$

BCV_i – earliest time (hours) to begin service of contractual voyage $i \in CV$ (after queue time)

BHI_v – minimum time (in hours) to hire-in ship $v \in VHI$

BSV_r – earliest time (hours) to begin service of spot voyage $r \in SV$ (after queue time)

CV – set of contractual voyages

CVP_i – set of port calls of voyage $i \in CV$

DP_{pq} – distance (nautical miles) between port $p \in P$ and port $q \in P$

ECO_v – maximum time (in hours) to charter ship $v \in VCO$

ECV_i – latest time (hours) to begin service of contractual voyage $i \in CV$

EHI_v – maximum time (in hours) to hire-in ship $v \in VHI$

ESV_r – latest time (hours) to begin service of spot voyage $r \in SV$

FC_v – fixed cost (US$/year) of ship $v \in V$

FHI_v – fixed cost (US$/year) of ship $v \in VHI$

latestTime – the latest time of the planning horizon

LCV_{ip} – tons of cargo that must be loaded at port $p \in CVP_i$

$LPSV_r$ – loading port of spot voyage $r \in SV$

LSV_r – tons of cargo that must be loaded at port $LPSV_r$ and then unloaded at port $UPSV_r$ of spot voyage $r \in SV$

$LUCv$ – lay-up cost of ship $v \in VO$

LUT_{vp} – loading and unloading times (hours/ton) of ship $v \in V$ at port $p \in P$

NT_v – net tonnage (capacity in tons) of ship $v \in V$

$o_i(a)$, $a \in \{1,\dots,|CVP_i|\}$ – a-iest port call of set CVP_i

P – set of ports

PA_v – port $p \in P$ in which ship $v \in V$ will be available for its first voyage

Q_{vp} – queue and loading/unloading preparation times (hours) of ship $v \in V$ at port $p \in P$

RC_{vp} – running cost (US$/hour) of ship $v \in V$ at port $p \in P$

RCO_v – revenue (US$) obtained if ship $v \in V$ is chartered

RCV_i – revenue (US$) obtained if contractual voyage $i \in CV$ is served

RSV_r – revenue (US$) obtained if spot voyage $r \in SV$ is served

SP_v – sailing speed (in knots) of ship $v \in V$

SV – set of spot voyages

TA_v – time (hours) in which ship $v \in V$ will be available for its first voyage

TCV_i – latest time (hours) to complete contractual voyage $i \in CV$

TSV_r – latest time (hours) to complete spot voyage $r \in SV$

UCV_{ip} – tons of cargo that must be unloaded at port $p \in CVP_i$

$UPSV_r$ – unloading port of spot voyage $r \in SV$

V - set of ships

VC_v – variable cost (US$/nautical mile) of ship $v \in V$

VCO - set of own ships that may be chartered

$VCHI_v$ – variable cost (US$/hour) of ship $v \in VHI$

VHI – set of hire-in ships

VLU – set of own ships that may be laid-up

VO – set of own ships

$VRCO_v$ – variable revenue (US$/hour) obtained if ship $v \in VCO$ is chartered

Nodes and mathematical model:

BC_n – set of indexes $i \in CV$ of contractual voyages of node $n \in N$

BS_n – set of indexes $r \in SV$ of spot voyages of node $n \in N$

BT_{vn} – earliest time (hours) for ship $v \in V$ to start service at node $n \in N$

$cap_{vn} \in R^+$ – decision variable for available net tonnage capacity of ship $v \in V$ just before starting voyage node $n \in N$

$CapMin_n$ – minimum idle capacity (tons) that a ship must have in order to serve node $n \in N$

$coph_v$ = 1 if ship $v \in VCO$ is chartered for the whole planning horizon, 0 otherwise.

coa_v = 1 if ship $v \in VCO$ is chartered after $endTime_v$, 0 otherwise.

cob_v = 1 if ship $v \in VCO$ is chartered before $startTime_v$, 0 otherwise.

$Demand_n$ – total tons that a node $n \in N$ loads (positive number) or unloads (negative number) to a ship

$endTime_v \in R^+$ – end time of the last port call of ship $v \in VCO \cup VHI$ during the planning horizon

$EstimatedProfit_{vn}$ – profit estimate if ship $v \in V$ serves node $n \in N$ and also serves all the other necessary spot voyage port calls (just before or just after) to maintain route feasibility

ET_{vn} – latest time (hours) for ship $v \in V$ to start service at node $n \in N$

M_1, M_2, M_3, M_4 and M_5 – large numbers for the MIP model

N – set of nodes (types 1, 2 or 3)

$o_n(a)$, $a \in \{1,...,|Ports_n|\}$ – a-iest port call of set $Ports_n$

$Ports_n$ – ordered sequence of port calls of node $n \in N$

$Profit_{vn}$ – profit (US$) obtained if ship $v \in V$ serves node $n \in N$

$st_{vn} \in R^+$ – decision variable for start time of voyage node $n \in N$ for ship $v \in V$

$startTime_v \in R^+$ – start time of first port call of ship $v \in VCO$ during the planning horizon

top_1 – number of best nodes to select considering the triplet "contractual voyage x subset of spot voyages x ship" (MIP best nodes solution method)

top_2 – number of best nodes to select considering the pair "contractual voyage x ship" (MIP best nodes solution method)

$TotalTime_{vn}$ – total time (hours) that a ship $v \in V$ would spend in node $n \in N$

$type_n$ – type $\in \{1, 2, 3\}$ of node $n \in N$

x_{vmn} = 1 if ship $v \in V$ services voyage node m $\in N$ just before voyage node $n \in N$, 0 otherwise.

$xfirst_{vn}$ = 1 if voyage node $n \in N$ is the first node of ship $v \in V$, 0 otherwise.

$xlast_{vn}$ = 1 if voyage node $n \in N$ is the last node of ship $v \in V$, 0 otherwise.

z_v = 1 if ship $v \in V$ is used during the planning horizon, 0 otherwise

Tabu search symbols:

$currentIter$ = count of the current iteration

$currentNumberOfNeighbors$ = number of neighboring solutions found at current iteration

$dist_{(s1,s2)}$ = distance measure between solutions s_1 and s_2

$distAvg$ = average distance of the pool defined as $distAvg = \frac{\sum_{i,j \in pool, i \neq j} dist_{(s_i, s_j)}}{poolMax \times (poolMax - 1)}$

distMax = maximum distance of the pool defined as

$$distMax = max\left(dist_i = min\left(dist_{(s_i,s_j)}: \forall\, j \in pool, i \neq j\right) \forall\, i \in pool\right)$$

dist_perc = percentage (between 0% and 100%) of the set of the total neighboring solutions that should be fully evaluated when using the list of candidates

$ExceededTime_r$ = sum of all time window violations of a route $r \in R$

$frequency_r$ = the sum of the corresponding values of $voyageIsTheFirstOfRoute(i)$ and $voyage2AfterVoyage1(i,j)$ of route r

iteration = a move from the current solution to another neighboring solution

iterIncumbentWasFound = iteration in which the incumbent solution (best feasible solution found so far) was found

minNumberOfNeighbors = parameter used to reduce *tabuTenureToAdd* if *currentNumberOfNeighbors* < *minNumberOfNeighbors*

$maxFrequency = max\left(voyageIsTheFirstOfRoute(i), voyage2AfterVoyage1(i,j)\right),\ \forall i,j \in |CV|$

maxIter = estimate of the number of iterations to be performed by Tabu Search

maxNumberOfNeighbors = parameter used to increase *tabuTenureToAdd* if *currentNumberOfNeighbors* > *maxNumberOfNeighbors*

maxPositiveProfitIncrease = maximum non-negative profit increase between two iterations during the past 20 iterations

Pen = non-negative penalty parameter for time window violation

phase3_max_restarts = maximum number of TS restarts from the pool during Phase 3

phase4_max_restarts = maximum number of TS restarts from the pool during Phase 4

poolMax = maximum number of solutions of the pool

R = set of ship routes of a solution

$route\_profit_r$ = profit of a route $r \in R$

Solution_Profit = total profit of a feasible or infeasible solution

t_1 = if a contractual voyage *i* was moved from ship *a* to ship *b*, voyage *i* must remain on ship *b* (is tabu) for at least t_1 iterations

t_2 = if a contractual voyage *i* was moved from ship *a* to ship *b*, voyage *i* must not return to ship *a* (is tabu) for at least t_2 iterations ($t_2 > t_1$)

t_3 = if a swap move was executed between ships *a* and *b*, another swap move between ships *a* and *b* may only occur (is not tabu) after t_3 iterations *tabuTenureToAdd* = variable that may assume a value between -10 and 10 that is used to adaptively adjust t_1, t_2 and t_3 as a function of *currentNumberOfNeighbors*

totalCalcs = parameter of the maximum number of $route\_profit_r$ calculations during the search (stop criterion)

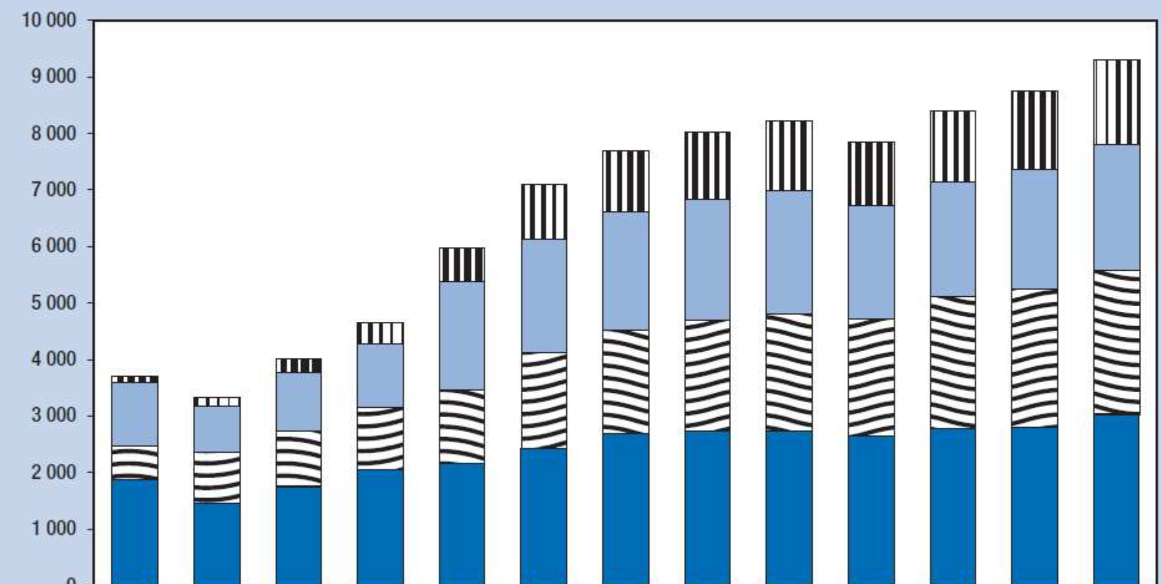
voyageIsTheFirstOfRoute(i) = number of times that voyage *i* is the first voyage of the route considering all solutions of the pool

voyage2AfterVoyage1(i,j) = number of times that voyage *j* is served immediately after voyage *i* considering all solutions of the pool

SUMMARY

1. INTRODUCTION

Maritime transportation is the backbone of international trade and has been experiencing a rapid growth over the past decades. The volume of transported cargo is estimated to have reached more than 9 billion tons of loaded cargo in 2012 (UNCTAD, 2012)[1]. Figure 1 shows that the sea born trade has more than doubled since 1980, with a significant growth in the five major bulks (iron ore, grain, coal, bauxite/alumina and phosphate), other dry cargo and containerized cargo.



	1980	1985	1990	1995	2000	2005	2006	2007	2008	2009	2010	2011	2012
Container	102	152	234	371	598	969	1 076	1 193	1 249	1 127	1 275	1 385	1 498
Other dry cargo	1 123	819	1 031	1 125	1 928	2 009	2 112	2 141	2 173	2 004	2 027	2 090	2 219
Five major bulks	608	900	988	1 105	1 295	1 709	1 814	1 953	2 065	2 085	2 335	2 477	2 547
Oil and gas	1 871	1 459	1 755	2 050	2 163	2 422	2 698	2 747	2 742	2 642	2 772	2 796	3 033

Figure 1. International seaborne trade in millions of tons loaded, UNCTAD (2012)

Ships are extremely expensive resources and the cost of single newly built ship ranges from US$ 10 million (a 500 TEU[2] container ship) to US$ 208 million (an LNG – Liquefied Natural Gas – carrier of 160.000 m^3), as shown in Table 1. When a company invests in a fleet of ships, either newly build or second-hand ships, the company expects high fleet utilization and occupation in order to increase its revenue. Thus, the need of an effective use of the fleet, either owned by the company or hired through short or long term contracts, compels planner professionals of the shipping companies to search for optimized fleet sizing, routing and scheduling solutions. In addition, since ship investments reach around the millions of dollars and

[1] United Nations Conference on Trade and Development

[2] Twenty-foot equivalent unit, for example, a standard container with 6.10 meters long x 2.44 meters wide x 2.59 meters high.

ship daily costs amounts to tens of thousands of dollars, a relatively small percentage improvement in the projected costs of a ship route may result in very large financial gains.

Table 1. Average price of a newly built ship in millions of US$, UNCTAD (2011)

Type and size of vessel	2003	2004	2005	2006	2007	2008	2009	2010	Percentage change 2010/2009
Oil tanker – Handy, 50 000 dwt	28	35	42	47	50	52	40	36	-10.0
Oil tanker – Suezmax, 160 000 dwt	47	60	73	76	85	94	70	66	-5.7
Oil tanker – VLCC, 300 000 dwt	67	91	119	125	136	153	116	103	-11.2
Chemical tanker – 12 000 dwt	12	16	18	21	33	34	33	28	-15.2
LPG carrier – 15 000 m³	28	36	45	49	51	52	46	41	-10.9
LNG carrier – 160 000 m³	153	173	205	217	237	222	226	208	-8.0
Dry bulk – Handysize, 30 000 dwt	16	19	21	22	33	38	29	25	-13.8
Dry bulk – Panamax, 75 000 dwt	23	32	35	36	47	54	39	35	-10.3
Dry bulk – Capesize, 170 000 dwt	38	55	62	62	84	97	69	58	-15.9
Container – geared, 500 TEUs	13	18	18	16	16	21	14	10	-28.6
Container – gearless, 6 500 TEUs	67	86	101	98	97	108	87	75	-13.8
Container – gearless, 12 000 TEUs	n.a.	n.a.	n.a.	n.a.	154	164	114	107	-6.1

The benefits that may be captured by the potential saving opportunities in maritime transport have attracted the attention to better decision making supported by operations research algorithms and decision support systems. These benefits have also encouraged the development of this present study.

A common classification of shipping companies in modes of maritime transportation is liner, tramp and industrial operations (Lawrence, 1972). A liner shipping company operates similar to a bus line, following a determined and published route. The liner company pickups and delivers client cargoes, e.g. containers, along the route analogous to the hop-on and hop-off of passengers in a bus line. A tramp shipping company does not have a predefined route to follow, the route is constructed and executed as new transport demands, such as dry bulk, gas or chemicals arrive. The analogy here is that a tramp company operates like a taxicab, picking up and delivering passengers while it is on the way. The tramp shipping operation may be full shipload, as of a taxicab with a single passenger, or parcel loads, as of a shared taxicab. The industrial operation is common for a verticalized company, such as mining companies that own or control both the cargo and the ships. The goal of the industrial shipping company is to move its own cargo to determined destinations at a minimum cost with its own or dedicated hired fleet. Table 2 presents additional information on liner, tramp and industrial operations.

Table 2. Description of the liner, tramp and industrial operations

	Liner	Tramp	Industrial
Business model	•The company establishes trade routes and frequency of service (similar to bus itineraries) •The regional offices sell available capacity to shippers, usually yearly contracts, with a few months in advance. These contracts, once accepted, are mandatory to be served and may represent from 80% to 95% of the liner company's business •The company attempts to fill idleness with spot demand	•The company (usually) owns a small or a medium fleet of ships and sells its capacity in two-ways: ▸ Contracts of affreightment: commitment to move a certain amount of cargo from Port 'A' to Port 'B' within a determined time window. May require one or more voyages and could be full or parcel shipload ▸ Spot voyages: commitment to serve a voyage from Port 'A' to Port 'B' (similar to a taxi cab, usually full shipload)	•An industrial company has a high volume of liquid or dry bulk cargo to transport, such as oil, chemicals, minerals or grains •Because of the high volume, the company decides to own or control a fleet of ships, including decisions of ship design, number of ships, routes and scheduling •The company often faces supply chain decisions, considering the integrated planning of production, inventory and port operations
Cargo owner	Shipper (client of the liner company)	Shipper (client of the tramp company)	Industrial company
Ship owner	•Liner company •May hire in other ships (time charter contract) if profitable or if necessary to fulfill mandatory contracts •May charter out, lay-up or scrap excess ships	•Tramp company •May hire in (time charter contract) if profitable to accept contracts of affreightment or to accommodate a peak of spot voyages demand •May charter out, lay-up or scrap excess ships	•Industrial company •If possible, the company sizes the number of ships below its long term needs and hire in (time charter contract from tramp companies) to accommodate peak demand
Financial objective	To maximize profit	To maximize profit	To minimize overall supply chain and transportation costs
Key decisions	•Trade routes and itinerary •Transshipment of containers •Positioning of empty containers •Fleet mix and deployment •Scheduling of spot voyages •Balanced ship loading	•Fleet mix and deployment •Ship routing and scheduling •Mix between contracts of affreightment and spot voyages (decision mainly depends on the estimate of the future price of the spot voyage market)	•Fleet mix and deployment •Ship scheduling •Integrated supply chain planning •Inventory and environmental routing to fulfill the supply chain plan (e.g. avoid product stock-out)
Type of ships and cargo	•Containers •General cargo (e.g. packaged goods in ship's hold or deck) •Refrigerated loads •Roll-on-Rool-off (Ro-Ro ramps for trucks/cars)	•Dry bulk (e.g. grain, coal, iron ore and other minerals) •Tankers (crude oil, liquids in bulk) •Liquefied gas •Refrigerated ships	•Dry bulk •Tankers •Liquefied gas •Specialized cargo with specific packaging and/or handling equipment
Size and frequency of shipments	•Parcel shiploads: large number of port calls and clients per voyage (e.g. container shipment) •Routes may be served weekly (or daily) and through more than one ship (e.g. a traderoute with a 4 week duration and a weekly frequency requires at least 4 ships)	•Full shipload: a pick-up and delivery from a single port of origin to a single port of destination •Parcel shipload: few clients and port calls per voyage •Frequency depends on the volume of the contracts of affreightment	•Full shipload (usually): a pick-up and delivery from a single port of origin to a single port of destination •Frequency depends of the integrated supply chain planning

These three operations face common and specific planning problems which vary according to the length of the planning horizon, to the commitment created by the decision making process and to the uncertainty about the future. Table 3, inspired on the list of problems of Christiansen et al. (2007), shows some of the main problems in the planning of maritime transport and suggests a relative importance of each problem to each maritime operation, namely, liner, tramp, industrial. The relative importance should be interpreted as an ordinary indication for a general case and may be changed according to one company specific environment.

Table 3. Summary of maritime problems and relative importance

	Example of problems	Liner	Tramp	Industrial
Strategic	•Market and trade selection ▸ Which markets and geographies should be serviced?	+++	+++	++
	•Ship design ▸ What is the optimal size of the ship? ▸ What on board loading/unloading equipment are needed?	++	++	+++
	•Network and transportation system design ▸ Are hubs and transshipment ports desirable? ▸ Are there intermodal (rail, road, barge) integrated services? •Fixed route/itinerary determination (e.g. trade routes) ▸ What are the port calls and frequency of service of each trade route?	+++	-	++
	•Fleet size and mix decisions ▸ How many ships, and of which type, should be on the fleet? ▸ Should excess ships be scrapped?	+++	+++	+++
	•Contract evaluation ▸ Which long term contracts should be taken? ▸ How can a company be hedged against spot market price change?	++	+++	-
	•Port/terminal location, size and design •Supply chain planning ▸ How is shipping affected by, and how it affects, production, inventory and other integrated processes of the supply chain?	-	-	+++
Tactical	•Fleet size and mix decisions ▸ Are changes on current fleet size/mix desirable? ▸ Which hired and/or chartered contracts should be engaged?	+++	+++	+++
	•Contract evaluation ▸ Which short term (spot) contracts should be taken?	++	+++	-
	•Fleet deployment ▸ How many and which ships should serve which trade routes?	+++	-	-
	•Ship routing ▸ What is the best sequence of port calls for each ship?	++	+++	+
	•Ship scheduling ▸ When should each ship start/end service at each port call?	+++	+++	+++
	•Ship refueling ▸ When and in which port should each ship be refueled?	++	+++	++
	•Inventory ship routing ▸ What should the ship route and scheduling be so to maintain inventory levels within the desired interval?	-	+	+++
Operatio-nal	•Cruising speed selection ▸ What is the optimum speed that minimizes fuel consumption and also services all port calls within time windows?	++	++	++
	•Ship loading ▸ How should the cargo (e.g. containers) be placed inside the hold of the ship and also above the deck to maintain balance?	+++	+	+
	•Environmental routing ▸ How should the routes be constructed considering waves, tides, currents and bad weather forecasting?	++	+	+++

Problem relative importance: - Practically none + Low ++ Medium +++ High

Problems approached in this work

It is also worthwhile to highlight the main differences between vehicle routing problems and maritime problems. For instance, Ronen (1983, 1993) and Christiansen et al. (2004) mention that ship problems differ from vehicle problems in the following aspects:

- There is no depot and ships do not necessarily return to their origin;
- Ships are operated around the clock while vehicles are operated, usually, only during the day. Therefore, maritime operations do not have a time buffer to withstand delays, whereas delays in road transportation could be compensated through a shorter night break;
- Because of the longer duration of the voyages, there is more uncertainty in the scheduling of ships than that of vehicles;
- Ships are different from each other in their cost structure not only because of different capacities, equipment and sailing speed, but also because of fluctuations in the ship and time charter markets;
- Ships pay port fees and operate mostly in international trade.

This work proposes a mixed integer programming (MIP) formulation and a tabu search for the Fleet Size and Mix Routing Tactical Problem (FSMRTP) in liner shipping that could also be extended to tramp shipping operations. For the medium term planning horizon, for example from six months to a year, fleet size decisions determine the necessary number of ships and select the types of ships to use, including the evaluation of the following alternatives: to lay-up a ship, i.e., to moor it in a protected anchorage or berth with most onboard systems shut down to reduce costs, to hire or to charter that means hiring a ship from another company or renting a ship to another company, respectively, for a determined period of time and price. Therefore, the objective of the FSMRTP is to maximize profit through the determination of: (i) the number and type of its own ships, as well as the number and type of laid-up and hired and chartered ships, (ii) the set of spot voyages to be served during the planning horizon and (iii) the ship routes and schedules.

The robustness of the solution of the FSMRTP depends on the level of uncertainty of future demands and of the future price of the charter market. Since the uncertainty is proportional to the length of the planning horizon, the planning horizon for the FSMRTP should not exceed, for example, one year. The next section presents the detailed description of the FSMRTP.

1.1. PROBLEM DESCRIPTION

The FSMRTP consists of the deployment of ships to trade routes, the routing of contractual and spot voyages, and the scheduling of the ships. Changes in fleet size and mix are allowed, such as to hire a ship to accommodate a peak of demand or to charter a ship for the remaining of the year if the overall fleet utilization is too low. A solution of the FSMRTP is feasible if all contractual voyages of the planning horizon are served without exceeding ship's capacity and violating voyages time window constraints. Spot voyages may be served if feasible and profitable. A solution of the FSMRTP is optimal if it is feasible and if it maximizes overall profit, which is defined as the sum of total revenues (charter, contractual and spot voyages revenues) minus the sum of total costs (fixed, variable, lay-up, hire and operating costs).

Four problem models, presented in Table 4, are proposed to address the FSMRTP. The SIMPLE problem models only consider owned or controlled ships and these ships may not be either laid-up or chartered. There are two models of SIMPLE, one that serves only contractual voyages (SIMPLE.Cv) and one that serves contractual voyages and may serve spot voyages (SIMPLE.CvSv). The FULL problem models consider owner's ships that may be laid-up and/or chartered, and hired ships. There are also two models of FULL, one that serves only contractual voyages (FULL.Cv) and one that serves contractual voyages and may serve spot voyages (FULL.CvSv).

Table 4. Problem models

Model	Voyages		Ships		
	Contractual	Spot	Lay-up	Charter	Hire
SIMPLE.Cv	✓				
SIMPLE.CvSv	✓	✓			
FULL.Cv	✓		✓	✓	✓
FULL.CvSv	✓	✓	✓	✓	✓

1.1.1. Ships and voyages

Ships have different capacities and may transport different cargo types. Sailing, loading and unloading times may vary among ships and according to the assignment of ships to cargoes. Ships with similar characteristics, such as cargo type, loading/unloading equipment and sailing time, define a ship class.

To solve any of the four problem models, ships must be assigned to voyages. A voyage is defined as:

a) a number of port calls to pick up cargoes;

b) a number of port calls to deliver cargoes;

c) a fixed quantity of cargo to be picked up and delivered in each port call;

d) a time window to start the first port call;

e) a time window to finish the last port call which can be based on an estimate of the duration of the voyage.

Each port call has a queue and loading/unloading times. To evaluate if a ship has started service at a port within the time window constraint, the queue time is added to the arrival time to compare with the time window range. Loading and unloading times may vary according to the type of ships and volume of cargoes.

The waiting time of a ship is the time between its arrival and the start of unloading, and it depends on the queue of ships in a port. Even though large waiting times reduce profit, there is not a constraint for the maximum waiting time of a ship in any port call.

The volume of cargo on a ship after each port call cannot exceed the ship capacity. To evaluate these capacity constraints, it is assumed that when a ship reaches a port call, the unloading operation is performed before the loading operation.

A voyage may be also considered as an instance of a trade route. Liner shipping companies make public the set of trade routes to the shipping market and then sell contractual voyages of each trade route. Figure 2 shows an example of a trade route that may load cargo at Itajaí, Santos and Suape ports and then unload cargo at Rotterdam and Hamburg ports.

Figure 2. Example of a trade route

Usually, there is a predetermined frequency of service for each trade route, such as daily, twice a week, weekly, and regional offices of the shipping company sell the transportation capacity to clients with yearly contracts. After the contracts have been signed, the shipping company knows the start and end date of voyages and has an estimate of how much cargo must be loaded and unloaded at each port. The certainty of this estimate varies according to the type of clients and of the length of the planning horizon.

Thus, a trade route with a defined time window and the volume of each cargo type is a voyage. Following the example of Figure 2, the shipping company would first determine that a ship will start this trade route between May 1 and May 3 and that this ship will reach the Hamburg port no later than June 4. At this time, the trade route becomes a voyage and a ship must be assigned to this voyage.

There may be restrictions on the assignment of ships to voyages, for example:

a) Type of ships: some cargoes cannot be assigned to certain type of ships;
b) Port restrictions: draft constraints and requirements of loading/unloading equipment;
c) Contractual obligations: client contracts can determine the type and size of ships that may transport the cargo.

After client contracts have been signed for a specific voyage, other spot voyages with a single port to load cargo and another port to unload cargo (e.g. to transport 5.000 tons from Santos to Rotterdam) may be accepted by the shipping company if the overall profit is increased and if there is enough capacity at the ship. Spot voyages revenue may represent about 5% to 30% of the total revenues of the shipping company.

1.1.2. Costs and revenues

The shipping company may operate with own and/or hired ships and may also charter additional ships. A different cost structure occurs in each case.

A fixed cost is incurred if the owner's ship is used during the planning horizon to serve at least one voyage or if an own ship is chartered. Depending on the charter market, the charter revenue may be smaller or larger than this fixed cost. To charter a ship is similar to renting an asset, in which there is a minimum and maximum renting time, a fixed rent income which contributes to cover the fixed costs, and a variable renting income proportional to the amount of the renting time. The duration of the charter contracts is limited to given lower and upper bounds for the contract period and can be at most as large as the whole planning horizon period. In addition, it is assumed that each ship may be engaged into at most two non-simultaneous charter contracts during the planning horizon (e.g. for two different companies). In the case in which more than two contracts are needed to be modeled, the lower and upper bounds for the contract period could be changed to accommodate a larger period of time of multiple contracts.

If the shipping company has no use for some of its ships for a long period of time, an option is to lay-up ships to reduce the fixed cost. Lay-up is to moor a ship at a protected anchorage or berth for a period of time of at least 5-6 months with most onboard systems shut down. This operation decreases fixed and insurance costs, reduces wear and tear of the ship and of the machinery, and may be also combined with maintenance operations. Because of the financial and time commitment of laying-up a ship, the options of laying-up and chartering are considered mutually exclusive for the same ship.

The shipping company may also hire ships to complement its own fleet. A ship could be hired in for a period of time between given lower and upper bounds of the duration of the contract. A fixed cost and variable cost proportional to the hire period is incurred. It is assumed that only a single hire contract may be settled for each ship during the planning horizon. Similar to the charter assumptions, longer hire periods may be modeled by changing the bounds.

In summary, the costs associated to ships are:

- Fixed costs ($/year): personnel, supplies, equipment, maintenance, repair, administration (e.g. insurance, office overhead, agency fees), cost of capital (e.g. financing, leasing), make ready to sail costs and all other running costs associated to keep the ship operational that do not depend on the distance travelled;
- Lay-up costs ($/year): administration, cost of capital and lay-up service and maintenance costs;
- Hire fixed costs ($/contract) and variable costs ($/day or $/hour): renting costs to cover administration, cost of capital and profit of the owner of the ship;
- Variable costs:
  - Daily running costs at ports ($/hours in port): port charges and fuel to maintain ships at port;
  - Fuel costs ($/nautical mile): fuel for ballast, parcel and full shipload sailing.

Although fuel cost is approximately proportional to the third power of the speed and, sometimes, even to the amount of cargo on the ship, this study considers that ships have different speeds and that the speed of each ship is fixed and given. This seems as a reasonable assumption for strategic and tactical planning problems.

Table 5 summarizes the options that a shipping company has for own and hired ships and the associated costs and revenues of each decision.

Table 5. Costs and revenues associated with each ship mix decision

Does the shipping company own the ship?	Was the ship used to serve the voyages?	Decision during the planning horizon	Associated Costs	Associated Revenues
Yes	Yes	**Use the ship**	•Fixed •Variable	•Voyages
	No	**Charter out the ship**	•Fixed	•Charter out fixed and variable
	Yes	**Both use and charter out the ship**	•Fixed •Variable	•Voyages •Charter out fixed and variable
	No	**Lay-up the ship**	Lay-up costs	-
	No	**Do not use the ship**	•None[a]	-
No	Yes	**Hire in the ship**	•Hire in fixed and variable costs	•Voyages
	No	**Do not use the ship**	-	-

[a] *If the company has a fixed cost for a certain unused ship, this cost could be set as a lay-up cost.*

1.1.3. Simplifying assumptions

This research encompasses most of real-life parameters, constraints and objectives. Still, some simplifying assumptions are considered to reduce combinatorial complexity. The assumptions below may be further addressed to bring problem solutions closer to reality.

a) Ships sail on a fixed speed and no slow steaming is allowed: since fuel consumption per nautical mile increases with speed, a ship could operate at lower speeds to reduce variable costs;

b) Ballast, full load and restricted water speeds are the same for each ship during the whole voyage: in reality, these speeds are different and total sailing time is a composition of the number of nautical miles sailed on each speed;

c) Ships have a single capacity constraint (e.g. one single compartment): some ships have several compartments with different capacities each;

d) Ships are available during the whole planning horizon with no interruption for maintenance: maintenance and out-of-service times are scheduled in advance for each ship and usually some ship will not be available during the whole planning horizon. This may be modeled as a "dummy" port call;

e) Inventory costs of goods in transit are not considered: each day that the cargo is on board has a cost for the company that owns the cargo. Thus, the price of a voyage could be a function of the voyage duration.

The next section presents how these assumptions were applied to create the set of test problems.

1.1.4. Set of test problems

A random test problem generator that considers real-life assumptions and parameters was developed to create a set of 14 test problems for each problem model of Table 4, totaling 56 test problems. Most of the assumptions are based on information available on ship carriers' websites and the report of UNCTAD (2011).

The number of ships ranges from 18 to 50 owner's ships plus 6 to 32 hired ships, resulting in a total of 24 to 82 ships. Each ship belongs to a ship class, which determines most of ships parameters such as capacity, sailing speed and costs. Table 6 shows the 6 classes of ships that were considered in the data set.

Table 6. Types and capacities of ships

Ship class	Capacity (net tonnage)
Handysize1	10,000
Handysize2	20,000
Supramax	30,000
Panamax	40,000
Capesize1	70,000
Capesize2	90,000

The test problems were created with at least one ship of each ship class of Table 6. In the majority of the test problems, Supramax, Panamax and Capesize are the predominant ship classes of the fleet.

Each ship has a sailing speed and fixed and variable operating costs which are randomly selected according to the parameters of Table 7.

Table 7. Sailing speed and costs of ships

Class	Sailing Speed		Fixed costs (US$ if own ship is used during the next year)		Variable costs (US$/nautical mile)	
	From	To	From	To	From	To
Handysize1	15	16	3,000,000	4,500,000	20	30
Handysize2	15	16	4,200,000	5,000,000	20	30
Supramax	14	16	4,800,000	6,000,000	25	35
Panamax	14	15	5,200,000	6,400,000	25	35
Capesize1	13	15	8,500,000	10,000,000	30	40
Capesize2	12	15	9,500,000	11,500,000	30	40

If a ship is laid-up, the costs of Table 8 are applied. Table 9 shows the conditions of the charter and hire contracts.

Table 8. Ship lay-up costs

Class	Lay-up fixed costs (US$ if the ship is laid-up for the whole year)	
	From	To
Handysize1	300,000	1,800,000
Handysize2	420,000	2,000,000
Supramax	480,000	1,800,000
Panamax	520,000	1,920,000
Capesize1	850,000	2,500,000
Capesize2	950,000	2,875,000

Table 9. Charter and hire contracts assumptions

Ship Class	Fixed profit (US$ if the ship is chartered or hired)		Variable profit (US$/day)		Minimum number of days of contract		Maximum number of days of contract	
	From	To	From	To	From	To	From	To
Handysize1	381,819	2,127,273	6,384	17,736	60	120	150	300
Handysize2	534,546	2,363,637	8,928	19,704	60	120	150	300
Supramax	610,910	2,836,364	10,200	23,640	60	120	150	300
Panamax	661,819	3,025,455	11,040	25,224	60	120	150	300
Capesize1	1,081,819	4,727,273	18,048	39,408	60	120	150	300
Capesize2	1,209,091	5,436,364	20,160	45,312	60	120	150	300

Ships are available for the planning horizon at different times, because each ship must conclude the current service before it receives a new route assignment. The time at which an owner's ship and hired ships are available for planning are chosen randomly between 0 and 20 days and between 0 and 150 days, respectively.

A set of 53 worldwide ports is used to create the set of demands. A distance matrix between ports was created based on ports latitude and longitude. In addition, for each pair 'port *x* ship class', a service time, loading/unloading rate and running costs were determined randomly, as shown in Table 10.

Table 10. Service time, loading/unloading rate and running costs of port calls per ship class

Ship Class	Service Time of each port call (hours)		Loading/unloading rate (tons/hour)		Running Costs (US$/hour)	
	From	To	From	To	From	To
Handysize1	5	24	200	300	22	45
Handysize2	5	24	230	300	22	45
Supramax	8	36	250	600	23	75
Panamax	8	36	300	800	27	100
Capesize1	12	48	350	900	31	110
Capesize2	12	48	400	1000	35	115

The location of the ship at the time the ship is available is a port selected randomly from the set of 53 ports.

It may not be possible for a certain port to serve all types of ship class. For instance, a port may not be able to accommodate a large ship or the port may not have the adequate loading/unloading equipment. To simulate this constraint, service time is set to a maximum limit for some 'port *x* ship class' pairs (approximately 2% of all 318 pairs).

Each test problem has a set with 5 to 12 trade routes and each trade route has a maximum number of port calls between 15 and 30. The paths of the trade routes were based on the itinerary published at Saga Forest Carries website (SAGA, 2013) and are shown in Table 11.

Each problem of the data set has between 30 and 110 contractual voyages and between 10 and 34 spot voyages (around a 75 % ratio of contractual and spot voyages). The amount of cargo at each port call and voyages time window, duration and revenue are selected randomly following the parameters shown in Table 12.

Table 11. Set of trade routes and maximum number of port calls of each trade route

#	Trade Route Path		Maximum number of port calls
	From	To	
1	Europe	East coast of North America	15
2	Europe	East coast of South America	16
3	West coast of North America	Europe	15
4	East coast of South America	Europe	16
5	Far east	West coast of North America	26
6	Far east	East coast of South America	29
7	East coast of North America	Far east	28
8	East coast of South America	Far east (through the Atlantic ocean)	30
9	East coast of South America	Far east (through the Pacific ocean)	29
10	East coast of South America	West coast of North America	17
11	East coast of North America	East coast of South America	18
12	West coast of North America	East coast of North America	17

Table 12. Cargo of port calls, voyages time window, duration and revenue

	Contractual Voyages	Spot Voyages
Amount of cargo loaded/unloaded at each port call	from 400 to 30,000 tons	from 1,000 to 10,000 tons
Time window to start the voyage	3 to 10 days	5 to 15 days
Voyage duration (% of the time the slowest ship would take to complete the voyage)	between 95% and 115%	between 95% and 130%
Revenue if the voyage is served	from US$ 30/ton to US$ 100/ton	from US$ 40/ton to US$ 150/ton

An amount of cargo to load and to unload is assigned to each port call. This amount of cargo is chosen randomly in such a way that, in the end, the total amount of loaded cargo equals to the total amount of unloaded cargo.

Spot voyages have a single port call to load and a single port call to unload the cargo. The origin and destination port calls are chosen randomly from the set of 53 ports.

A voyage may not be served by all classes of ships (e.g. product incompatibility or commercial constraints). To simulate this constraint, there is a uniform probability of 4% of each ship class not serving each contractual or spot voyage. To guarantee the feasibility of serving a contractual voyage, the Capesize ship classes are always allowed to serve contractual voyages.

At last, the earliest time to start each voyage is chosen randomly so that the latest time is within the planning horizon.

1.2. SUMMARY OF PAPERS

The main part of this thesis is the result of two papers produced by the author and his PhD supervisor Vinícius Amaral Armentano. The first paper, ‘Fleet Deployment Optimization in Liner Shipping with Spot Voyages’, was submitted to the European Journal of Operational Research and presents the MIP model and the CPLEX results of the 4 problem models of the FSMRTP presented in Table 4. The second paper, ‘Tabu search for fleet deployment and routing in liner shipping’, was submitted to the Computers and Operations Research journal and describes the tabu search method applied to the same problems models.

The contribution of the first paper relies on a novel mixed integer programming model based on a direct graph with the minimum set of three types of non-dominated nodes. Then, exact and approximated methods are applied to the mathematical formulations of the 4 problem models. The formulations also include up to two charter contracts (or one hire contract) per ship and the revenues (or costs) of the contracts are proportional to their duration.

The research of the first paper started as a suggestion of Professor Vinícius Amaral Armentano from Unicamp. Two invited professors from the Norwegian University of Science and Technology, Kjetil Fagerholt and Lars Magnus Hvattum, contributed with a suggestion of reference papers and also with the distance matrix data between the 53 worldwide ports described in the previous section. Prof. Armentano presented the partial results of this paper at IFORS 2011 conference in Melbourne and MSc Branchini presented additional results at ISMP 2012 in Berlin.

The concept of the second paper began when the proposed MIP models proved to be inefficient to solve large instances of the problem. Therefore, the authors decided to implement heuristics methods such as tabu search. The proposed tabu search algorithm has several of the components described in Glover and Laguna (1997) and explores infeasible regions through insertion and exchange neighborhoods. The outline of the exploration of infeasible solutions is inspired on the application of another tabu search method, which was implemented by the authors in 2012, to the generalized assignment problem (Armentano and Branchini, 2013). Other contributions of this paper are the use of a candidate list to restrict the neighborhood size, the diversification through a tree search method and the improvement and management of a pool of elite solutions.

The remainder of this thesis is organized as follows. Next, section 1.3 presents some of the implementation issues concerning the programming of the MIP model and of the data structure. Section 1.4 contains the complete list of references. Subsequently, the first paper is presented in section 2 and the second paper is given in section 3.

1.3. IMPLEMENTATION ISSUES

This section describes some of the principles and ideas that were put into practice to improve the efficiency of the programming code. The majority of the comments presented here refer to data structures and considerations on how to insert the MIP model into CPLEX.

Some data structures were implemented to rapidly access the data that are stored in the memory. For example, the information associated with the MIP model variables is stored in a vector of variable objects. Since the models of the large test problems have more than one million variables, the computational time to search the information about a certain set of variables in a vector is very high. Therefore, the variables were classified into types, such as flow variables, time window variables, etc., and matrices were implemented to return the respective list of pointers to the variable objects. For instance, one matrix returns the list of pointers to the variable objects for every pair 'ship *x* type of variables' and another matrix returns the list of pointers for every pair 'node *x* type of variables'.

Numerical issues were encountered when implementing the MIP model. Such as, the CPLEX solver would label the test problem as infeasible when, in fact, the test problem was feasible. The numerical imprecision problems were solved with three main guidelines:

a) On every occasion that a large number, such as *Big Ms*, must be inserted into the model, the smallest valid large number is used. This avoided numerical errors especially when the large numbers are the coefficients of binary variables.

b) The tightest lower and upper bounds are always calculated and informed to the CPLEX model for the continuous variables.

c) The CPLEX tolerances, such as the tolerances to consider a number as an integer and to consider a constraint as violated, were manually adjusted.

Some values of the test problems have large magnitude such as millions or hundreds of thousands of dollars associated with voyages revenues and fixed costs of ships. It was also tested to insert these values in the mathematical model as millions of dollars, instead of dollars, to avoid large coefficients of the binary variables. However, this modification did not affect the overall performance. In this case, the default setting of the scaling parameter *ScaInd* of CPLEX guaranteed that the rows that contained such financial information were properly scaled by multiplying the rows of the model by suitable constants.

Whenever possible, the most frequent feasibility tests are executed only once and the results of these tests are stored in data structures to avoid repetitive computation. For example, the information on whether a ship may serve a contractual or spot voyage and on whether two contractual voyages may be served in sequence by each ship are stored in two vectors and one three dimensional matrix. The algorithms of both papers make use of these data structures.

In the tabu search algorithm, unnecessary repetitive calculations were also avoided with the use of memory and data structures related to the neighborhood moves. For example, the moves of the insertion, exchange and swap neighborhoods are stored, respectively, in three matrices: ‘contractual voyage *x* ship’, ‘contractual voyage *x* contractual voyage’ and ‘ship *x* ship’. Each matrix returns a pointer to a movement object that contains the absolute differences in profit, sailed distance, time window violation, and other information associated with the move, between the current solution and the resulting solution after the move. Therefore, the information on the quality and feasibility impact of the execution of a move is readily accessed. In addition, once a move is executed, only the movement objects associated with the affected ships and routes are recalculated.

At last, a particular attention was put on the testing of the algorithms. Routines were specially designed and implemented to test the consistency of the data structures, the behavior of the algorithms and the accuracy of the reported results. Because of the neighborhood data structures of the tabu search and the complexity of the MIP model, especially the combinatorial nature of the node model, a slower debug version of the algorithms, in which the test procedures are activated, was tested on a set of test problems of smaller size.

1.4. REFERENCES

Agarwal, R., & Ergun, Ö. (2008). Ship scheduling and network design for cargo routing in liner shipping. *Transportation Science*, 42(2), 175–196.

Álvarez, J. F. (2009). Joint routing and deployment of a fleet of container vessels. *Maritime Economics & Logistics*, 11, 186-208.

Armentano, V., & Branchini R. M. (2013). Meta-Heurísticas em Pesquisa Operacional. In H. S. Lopes, L. C. A. Rodrigues, & M. T. A. Steiner (Eds), *Uma Introdução à Busca Tabu* (pp. 33-52), São Paulo: Omnipax.

Bent, R. W., & Van Hentenryck, P. (2004). Scenario-based planning for partially dynamic vehicle routing with stochastic customers. *Operations Research*, 52, 977-987.

Besbes, O., & Savin, S. (2009). Going bunkers: the joint route selection and refueling problem. *Manufacturing & Service Operations Management*, 11(4), 694-711.

Brønmo, G., Christiansen, M., Fagerholt, K., & Nygreen, B. (2007a). A multi-start local search heuristic for ship scheduling – a computational study. *Computers and Operations Research*, 34(3), 900-917.

Brønmo, G., Christiansen, M., & Nygreen, B. (2007b). Ship routing and scheduling with flexible cargo sizes. *Journal of the Operational Research Society*, 58, 1167-1177.

Brønmo, G., Nygreen, B., & Lysgaard, J. (2010). Column generation approaches to ship scheduling with flexible cargo sizes. *European Journal of Operational Research*, 200(3), 139-150.

Cho, S. C., & Perakis, A. N. (1996). Optimal liner fleet routeing strategies. *Maritime Policy & Management*, 23(3), 249-259.

Christiansen, M., Fagerholt, K., & Ronen, D. (2004). Ship routing and scheduling – status and perspectives. *Transportation Science*, 38(1), 1-18.

Christiansen, M., Fagerholt, K., Nygreen, B., & Ronen, D. (2007). Handbooks in Operations Research and Management Science: Transportation. In C. Barnhart, & G. Laporte (Eds), *Maritime Transportation* (pp. 189-284). Amsterdam: North-Holland.

Christiansen, M., Fagerholt, K., Hasle, G., Nygreen, B., & Minsaas, A. (2009). Maritime transport optimization: An ocean of opportunities. *OR&MS Today*, 36(2), 26-31.

Christiansen, M., Fagerholt, K., Flatberg, T., Haugen, Ø., Kloster, O., & Lund, E. H. (2011) Maritime inventory routing with multiple products: A case study from the cement market. *European Journal of Operational Research*, 208, 86-94.

Christiansen, M., Fagerholt, K., Nygreen, B., & Ronen, D. (2013) Ship Routing and Scheduling in the New Millennium. *European Journal of Operational Research*, 228(3), 467-483.

Cordeau, J-F., Laporte, G., & Mercier, A. (2001). A unified tabu search heuristic for vehicle routing problems with time windows. *Journal of the Operational Research Society,* 52, 928-936.

Fagerholt, K. (2001). Ship scheduling with soft time windows – an optimization based approach. *European Journal of Operational Research*, 131(3), 559-571.

Fagerholt, K. (2004). A computer-based decision support system for ship fleet scheduling – Experience and future research. *Decision Support Systems*, 37(1), 35-47.

Fagerholt, K., Johnsen, T. A. V., & Lindstad, H. (2009). Fleet deployment in liner shipping–a case study. *Maritime Policy & Management*, 36(5), 397-409.

Fagerholt, K., Christiansen, M., Hvattum, L. M., Johnsen, T. A. V., & Vabø, T. J. (2010). A decision support methodology for strategic planning in maritime transportation. *OMEGA – The International of Management Science*, 38, 465-474.

Gelareh, S., & Meng, Q. (2010). A novel modeling approach for the fleet deployment problem within a short-term planning horizon. *Transportation Research Part E*, 46,76-89.

Glover, F., & Laguna, M. (1997). Tabu Search. Kluwer Academic Publishers.

Hennig, F., Nygreen, B., Christiansen, M., Fagerholt, K., Furman, K. C., Song, J., Kocis, G. R., & Warrick, P. H. (2012). Maritime crude oil transportation – A split pickup and split delivery problem. *European Journal of Operational Research*, 218, 764-774.

Hvattum, L.M., Løkketangen, A., & Laporte, G. (2007). A branch-and-regret heuristic for stochastic and dynamic vehicle routing problems. *Networks*, 49, 330-340.

Karlaftis, M. G., Kepaptsoglou, K., & Sambracos, E. (2009). Containership routing with time deadlines and simultaneous deliveries and pick-ups. *Transportation Research Part E*, 45, 210-221.

Kim, S-H., & Lee, K-K. (1997). An optimization-based decision support system for ship scheduling. *Computers Industrial Engineering*, 33(3-4), 689-692.

Korsvik, J.E., & Fagerholt, K. (2010). A tabu search heuristic for ship routing and scheduling with flexible cargo quantities. *Journal of Heuristics*, 16(2), 117-137.

Korsvik, J.E., Fagerholt, K., & Laporte, G. (2010). A tabu search heuristic for ship routing and scheduling. *Journal of the Operational Research Society*, 61, 594-603.

Korsvik, J.E., Fagerholt, K., & Laporte, G. (2011). A large neighbourhood search heuristic for ship routing and scheduling with split loads. *Computers & Operations Research*, 38, 474-483.

Lawrence, S. A. (1972). International sea transport: the years ahead. Lexington Books, Lexington, MA.

Meng, Q., Wang, T., & Wang, S. (2012). Short-term liner ship fleet planning with container transshipment and uncertain container shipment demand. *European Journal of Operational Research*, 223, 96-105.

Norstad, I., Fagerholt, K., & Laporte, G. (2011). Tramp ship routing and scheduling with speed optimization. *Transportation Research Part C*, 19, 853-865.

Ow, P.S, & Morton, T. E., (1988). Filtered beam search in scheduling. *International Journal of Production Research*, 26, 297-307.

Psarafatis, H. N., & Kontovas, C. A. (2013). Speed models for energy-efficient maritime transportation: A taxonomy and survey. *Transportation Research Part C*, 26, 331-351.

Ronen, D. (1983). Cargo ships routing and scheduling: Survey of models and problem. *European Journal of Operational Research*, 12, 119-126.

Ronen, D. (1993). Ship scheduling: The last decade. *European Journal of Operational Research*, 71(3), 325-333.

SAGA Forest Carriers website. (2013). http://www.sagafc.com.

Solomon, M. (1987). Algorithms for the vehicle routing and scheduling problem with time window constraints. *Operations Research*, 35(2), 254-265.

Tirado, G., Hvattum, L. M., Fagerholt, K., Cordeau, J-F. (2013). Heuristics for dynamics and stochastic routing in industrial shipping. *Computers and Operations Research*, 40(1), 253-263.

UNCTAD. (2011). Review of maritime transport. *United Nations Conference on Trade and Development*, New York and Geneva.

UNCTAD. (2012). Review of maritime transport. *United Nations Conference on Trade and Development*, New York and Geneva.

Yan, S., Chen, C-Y., & Lin, S-C. (2009). Ship scheduling and container shipment planning for liners in short-term operations. *Journal of Maritime Science Technology*, 14, 417-435.

2. PAPER A - FLEET DEPLOYMENT OPTIMIZATION IN LINER SHIPPING WITH SPOT VOYAGES

Fleet Deployment Optimization in Liner Shipping with Spot Voyages

Abstract

We address a tactical planning problem faced by many liner shipping companies that have committed contractual voyages while trying to serve optional spot voyages to increase its revenue over the medium-term horizon. The optimization problem is formulated as a mixed integer programming model that is defined on a directed graph whose nodes represent contractual and spot voyages. The decisions include the number and type of vessels deployed, the assignment of vessels to contractual and spot voyages and the determination of vessel routes and schedules in order to maximize profit. Computational results are reported.

Keywords: *Logistics, maritime transportation, liner shipping, routing, integer programming.*

1. Introduction

Maritime transportation is the backbone of international trade and has been experiencing a rapidly growth over the past decades. The volume of transported cargo is estimated to have reached almost 9 billion tons of loaded cargo in 2011 (UNCTAD[3], 2011). Ships are extremely expensive resources, and the cost of a single newly built ship may range from US$ 10 million (e.g. a 500 TEU container ship) to US$ 208 million, such as a Liquefied Natural Gas vessel of 160.000 m3 (UNCTAD, 2011). When a company invests in either newly built or second-hand ships, it expects high fleet utilization and occupation to obtain a higher revenue. Thus, the need of an effective use of the fleet, either owned by the company or hired/chartered through short or long term contracts, compels planner professionals of the shipping companies to search for optimized fleet sizing, routing and scheduling solutions. In addition, since ship investments reach around the millions of dollars and ship daily costs ranges tens of thousands of dollars, a relatively small percentage improvement in the projected costs of a ship route may result in large financial gains.

A common classification of shipping companies in modes of maritime transportation is liner, tramp and industrial operations (Lawrence, 1972). A liner shipping company operates similar to a bus line, following a determined and published route. The liner company pickups and delivers client cargoes, e.g. containers, along the route analogous to the hop-on and hop-off of

[3] United Nations Conference on Trade and Development.

passengers in a bus line. A tramp shipping company does not have a predefined route to follow, the route is constructed and executed as new transport demands, such as dry bulk, gas or chemicals arrive. The analogy here is that a tramp company operates like a taxi cab, picking up and delivering passengers while it is on the way. The tramp shipping operation may be full shipload, as of a taxi cab with a single passenger, or parcel loads, as of a shared taxi cab. The industrial operation is common on a verticalized company, such as mining companies that own or controls both the cargo and the ships. The goal of the industrial shipping company is to move its own cargo to determined destinations at a minimum cost with its own or dedicated hired in fleet. Table A.1 of Appendix A presents additional information on liner, tramp and industrial operations.

These three operations face common and planning problems, which vary according to the length of the planning horizon, to the commitment created by the decision making process and to the uncertainty about the future. Table A.2 of Appendix A, inspired on the list of problems of Christiansen et al. (2007), shows some of the main problems in the planning of maritime transport and suggests relative importance of each problem to each maritime operation, namely, liner, tramp, industrial. The relative importance should be interpreted as an ordinary indication for a general case and may be changed according to one company environment.

This work proposes a mixed integer programming (MIP) formulation for the Fleet Size and Mix Routing Tactical Problem (FSMRTP) in liner shipping which consists of committed contractual voyages. Optional spot voyages may also be served to increase the revenue over the medium term planning horizon of a few months up to one year. A voyage is a sequence of port calls with four attributes: time duration, profit, earliest and latest time to start the service and available capacity for the ship to serve the voyage.

Fleet size decisions determine the necessary number of ships and select the types of ships to use, including the evaluation of the following alternatives: to lay-up a ship, i.e., to moor it in a protected anchorage or berth with most onboard systems shut down to reduce costs, to hire a ship from another company or to charter a ship to another company, respectively, for a determined period of time and price. Therefore, the objective of the FSMRTP is to maximize profit through the determination of (i) the number and type of its own ships, as well as the number and type of laid-up, hired and chartered ships, (ii) the set of spot voyages to be served during the planning horizon and (iii) the ship routes and schedules.

The robustness of the solution of the FSMRTP depends on the level of uncertainty of future demands and the future price of the charter market. Since the uncertainty is proportional to the length of the planning horizon, the planning horizon for the FSMRTP should not exceed, for example, one year.

The main contribution of the proposed MIP model lies in a graph-based construction where nodes represent contractual and spot voyages, and routes corresponds to sets of arcs. The key point is to create the smallest set of feasible and non-dominated nodes with respect to the four attributes mentioned above. Dominated nodes are discarded, thereby eliminating poor quality feasible solutions of the problem. An approximated model that only considers the most promising nodes is also presented. In addition, the flexibility of the structure of the proposed MIP model is explored to address four different types problems in liner shipping. Another contribution is the manner in which the model handles the fixed and variable costs of the charter and hire contracts. For example, an own ship may be used to serve voyages and also may be chartered up to two contracts during the planning horizon. Also, the revenue and costs associated with the chartered and hired ships are proportional to the duration of the charter or hire contracts.

The remainder of the paper is organized as follows. Section 2 presents a literature review and Section 3 contains a detailed description of the problem. Section 4 shows the construction of nodes that represent contractual and spot voyages in a directed graph with arcs denoting sequences of voyages. Section 5 presents mixed integer programming models for four variants of the FSMRTP. Section 6 describes the computational experiments and, finally, conclusions are discussed in Section 7.

2. Literature review

This section presents some of the approaches used to model maritime fleet planning problems. There is a broad collection of models in the literature and each model emphasizes on different aspects of the problem. Moreover, some of these models have already been implemented into Decision Support Systems (DSS).

The first two surveys of Ronen (1983, 1993) deal with ship routing and scheduling, and related problems, such as fleet deployment, inventory routing and optimal cruising speed. In both articles, the author stresses the scarcity of published work in this area. The survey of Christiansen et al. (2004) divides the literature according to strategic, tactical and operational problems, and the three modes of operation in shipping: liner, tramp and industrial. The increasing research in maritime transportation is reflected in a much higher number of references, with the majority of problems based on real applications.

The work of Christiansen et al. (2007) is a comprehensive introduction to several aspects of maritime transportation, including the dependence of the world economy on worldwide trade and maritime characteristics, such as types of ships and cargo. In addition, the work discusses several planning problems and mathematical optimization models with an emphasis on ship routing and scheduling models.

Christiansen et al. (2012) provide a review on the research on ship routing and scheduling and also present four basic mathematical models: (i) network design and (ii) fleet deployment for liner shipping, (iii) cargo routing and scheduling and (iv) maritime inventory routing for tramp and industrial shipping. One of the main concluding remarks is the economic importance of the optimization of the network design and fleet deployment problems for a liner shipping company. Operational and demand uncertainties are mentioned as important obstacles when dealing with such problems.

The reduction of fuel consumption has always been a major concern of ship carriers. Christiansen et al. (2009) report that transport fuel efficiency has improved from 0.025 kg of fuel to transport one container one nautical mile on the voyage from Asia to Europe in 2007, compared to 0.200 kg in 1970. However, the research for improvements on fuel efficiency is still very important for the optimization of the cruising speed of ships. Psarafatis and Kontovas (2013) point out that speed is a key variable and provide a taxonomy and survey of speed models for energy-efficient maritime transportation. Increasing fuel prices, depressed market conditions and environmental issues associated with the contribution of international shipping of CO_2 emission of 2.7% of the global emission, has increased the emphasis on the optimization of ship speed.

Fagerholt (2001) formulates the ship scheduling problem with soft time windows and also calculates the optimal speed of each route in order to minimize soft time windows penalties and operating costs. The results show that controlled time window violation produces better schedules and lower overall costs.

Cho and Perakis (1996) develop a fleet size and route optimization model for a liner container shipping company. A number of candidate routes are generated a priori and then the problem is solved as a linear programming model. A mixed integer programming model is presented to assess investments to expand fleet capacity.

Agarwal and Ergun (2008) present an integrated model to solve simultaneously the network design, referred in the paper as ship scheduling, and the cargo routing problem for liner shipment, including the alternative of transshipment of containers which involves moving a cargo from one ship to another at an intermediate port. Three algorithms are developed: a greedy heuristic, a column generation-based algorithm and a Benders decomposition-based algorithm. Computational tests show that the greedy heuristic is fast and produces good quality solutions for small problems, and the column generation-based algorithm is suitable for medium problems, and the Benders decomposition-based algorithm is more robust and produces high quality solutions for large problems.

Besbes and Savin (2009) address the joint route selection and refueling problem for liner and tramp shipping considering a single ship. The liner refueling problem is formulated as a long term average stochastic dynamic program and the authors prove that the optimal refueling policy has a capacitated price-dependent buy-up-to form. The tramp refueling problem is combined with the route selection problem in the cases of uniform and non-uniform fuel prices. The authors present numerical results for a real-life problem of a tramp ship and three ports and suggest expanding the solution approach to a multiple-ship problem as a future work.

Yan et al. (2009) apply a Lagrangian-based algorithm to solve the ship scheduling and container shipment planning problem. The algorithm is tested in a major Taiwanese shipping company and produced results that are 16.69% better than those obtained manually.

Decision support systems and methodology have been developed to aid professionals of the maritime industry. For instance, Kim and Lee (1997) present the prototype MoDiSS (Model-based DSS in Ship Scheduling) for bulk trade companies. Fagerholt (2004) and later Christiansen et al. (2009) present Turborouter, a flexible decision support system that helps the planner to assign ships to cargoes. In addition to the optimization heuristics, the work focus on user interfaces and reports in order to help the planner to modify and interact with the presented solution.

Fagerholt et al. (2010) present a decision support methodology for strategic planning, including contract analysis and fleet mix and size decisions, in tramp and industrial shipping. To ensure good long term strategic decisions, the authors also consider a significant amount of details regarding short term decisions, such as ship routing and scheduling according to a rolling horizon. The methodology consists of an integrated simulation and optimization of four main processes: (i) determine a set of strategic decisions, (ii) create a set of scenarios, (iii) use simulation to evaluate each strategic decision for each scenario and (iv) analyze results to determine the effects of each strategic decision. To simulate process (iii), the optimization tool Turborouter is used together with Ms-Excel spreadsheets, see Fagerholt (2004). The methodology is applied to a case study of a Norwegian shipping company and the results of the computational experiments provided valuable decision support information, such as whether the company should negotiate a longer notice time of the contracts of affreightment.

Gelareh and Meng (2010) approach the fleet deployment problem of liner shipping operations within a short-term planning horizon. The authors present a mixed nonlinear programming formulation that includes the optimization of the speed in which each ship sails in each link between two port calls. The nonlinear problem is linearized and then solved through standard commercial solvers such as CPLEX. A solution of this problem assign ships (own or hired in) to cyclic routes to meet the predetermined service frequency of each route for the 90 working days planning horizon. Excess capacity may be chartered and the possibility of changing the service frequency of each route is tested. More recently, Meng et al. (2012) propose a two-stage stochastic integer programming model for a liner shipping planning problem with container transshipment and demand uncertainty. The solution approach, based on dual decomposition and Lagrangian relaxation, is applied to a real world example of eight ship routes and 36 ports.

Hennig et al. (2012) address the split pickup and split delivery problem for maritime crude oil transportation. The authors introduce a path flow model, in which paths represent pre-generated ship routes, to solve six realistic oil tanker routing and scheduling test problems for a fleet between 2 and 6 ships. The proposed model finds feasible solutions in a short time, but only small instances can be solved to proven optimality during the optimization time of 24 hours.

3. Problem description

The FSMRTP consists of the deployment of ships to trade routes, the routing of contractual and spot voyages, and the scheduling of the ships. Changes in fleet size and mix are allowed, such as to hire a ship to accommodate a peak of demand, or to charter a ship for the remaining of the year if the overall fleet utilization is too low. A solution of the FSMRTP is feasible if all contractual voyages of the planning horizon are served without exceeding ships capacity and violating voyages time window constraints. Spot voyages may be served if feasible and profitable. A solution of the FSMRTP is optimal if it is feasible and if it maximizes overall profit, which is defined as the sum of total revenues (charter, contractual and spot voyages revenues) minus the sum of total costs (fixed, variable, lay-up, hiring and operating costs).

Four problem models, presented in Table 1, are proposed to address the FSMRTP. The SIMPLE problem models only consider owned or controlled ships, and these ships may not be either laid-up or chartered. There are two models of SIMPLE, one that serves only contractual voyages (SIMPLE.Cv), and one that serves contractual voyages and may serve spot voyages (SIMPLE.CvSv). The FULL problem models consider owner's ships that may be laid-up, chartered and hired during the planning horizon ships. There are also two models of FULL, one that serves only contractual voyages (FULL.Cv) and one that serves contractual voyages and may serve spot voyages (FULL.CvSv).

Table 1. Problem models

Model	Voyages		Ships		
	Contractual	Spot	Laid-up	Chartered	Hired
SIMPLE.Cv	✓				
SIMPLE.CvSv	✓	✓			
FULL.Cv	✓		✓	✓	✓
FULL.CvSv	✓	✓	✓	✓	✓

3.1. Ships and voyages

Ships have different capacities and may transport different cargo types. Sailing, loading and unloading times may vary among ships and according to the assignment of ships to cargoes. Ships with similar characteristics, such as cargo type, loading/unloading equipment and sailing time, define a ship class.

To solve any of the four problem models, ships must be assigned to voyages. A voyage is defined as a:

a) number of port calls to pick up cargoes;

b) number of port calls to deliver cargoes;

c) fixed quantity of cargo to be picked up and delivered in each port call;

d) time window to start the first port call;

e) time window to finish the last port call which can be based on an estimate of the duration of the voyage.

Each port call has a queue and loading/unloading times. To evaluate if a ship has started service at a port within the time window constraint, the queue time is added to the arrival time to compare with the time window range. Loading and unloading times may vary according to the type of ships and volume of cargoes.

The waiting time of a ship is the time between its arrival and the start of unloading, and it depends on the queue of ships in a port. Even though, large waiting times reduce profit, there is not a constraint for the maximum waiting time of a ship in any port call.

The volume of cargo at a ship after each port call cannot exceed the ship capacity. To evaluate these capacity constraints, it is assumed that when a ship reaches a port call, the unloading operation is performed before the loading operation.

A voyage may be also considered as an instance of a trade route. Liner shipping companies make public the set of trade routes to the shipping market and then sell contractual voyages of each trade route. Figure 1 shows an example of a trade route that may load cargo at Itajaí, Santos and Suape ports and then unload cargo at Rotterdam and Hamburg ports.

Figure 1. Example of a trade route

Usually, there is a predetermined frequency of service for each trade route, such as daily, twice a week, weekly, and regional offices of the shipping company sell the transportation capacity to clients with yearly contracts. After the contracts have been signed, the shipping company knows the start and end date of voyages and has an estimate of how much cargo must be loaded and unloaded at each port. The certainty of this estimate varies according to the clients and length of the planning horizon.

Thus, a trade route with a defined time window and the volume of each cargo type is a voyage. Following the example of Figure 1, the shipping company would first determine that a ship will start this trade route between May 1 and May 3 and that this ship will reach the

Hamburg port no later than June 4. At this time, the trade route becomes a voyage and a ship must be assigned to this voyage.

There may be restrictions in the assignment of ships to voyages, for example:

a) Type of ships: some cargoes cannot be assigned to certain types of ships;
b) Port restrictions: draft constraints and requirements of loading/unloading equipment;
c) Contractual obligations: client contracts can determine the type and size of ships that may transport the cargo.

After client contracts have been signed for a voyage, other spot voyages with a single port to load cargo and another port to unload cargo (e.g. to transport 5.000 ton from Santos to Rotterdam) may be accepted by the shipping company if the overall profit increases and if there is enough capacity at the ship. Spot voyages revenue may represent from 5% to 30% of the total revenues of the shipping company.

3.2. Costs and revenues

The shipping company may operate with own and/or hired ships and may also charter excess ships. A different cost structure occurs in each case.

A fixed cost during the planning horizon is incurred if the owner's ship is used to serve at least one voyage or if an own ship is chartered. Depending on the charter market, the charter out revenue may be smaller or larger than this fixed cost. Chartering a ship is similar to renting an asset for a minimum and a maximum time, a fixed rent income which contributes to cover the fixed costs, and a variable renting income proportional to the amount of the renting time. The duration of the charter contracts is limited to given lower and upper bounds for the contract period and can be at most as large as the whole planning horizon period. In addition, it is assumed that each ship may be engaged into at most two non-simultaneous charter contracts during the planning horizon. In the case in which more than two contracts are needed to be modeled, the lower and upper bounds for the contract period could be changed to accommodate a larger period of time of multiple contracts.

If the shipping company has no use for some of its ships for a long period of time, an option is to lay-up ships to reduce the fixed cost. Lay-up is to moor a ship at a protected anchorage or berth for a period of time of at least 5-6 months with most onboard systems shut down. This operation decreases fixed, and insurance costs, reduces wear and tear of the ship and machinery, and may be also combined with maintenance operations. Because of the financial and time commitment, the options of laying up and chartering are considered mutually exclusive for the same ship.

The shipping company may also hire ships to complement its own fleet. A ship could be hired for a period of time between given lower and upper bounds of the duration of the contract. A fixed cost and variable cost proportional to the hire period are incurred. It is assumed that only a single hire a contract may be settled for each ship during the planning horizon. Similar to the charter assumptions, longer hire periods may be modeled by changing the bounds.

In summary, the costs associated to ships are:

- Fixed costs ($/year): personnel, supplies, equipment, maintenance, repair, administration (e.g. insurance, office overhead, agency fees), cost of capital (e.g. financing, leasing), make ready to sail costs and all other running costs that do not depend on the distance travelled;
- Lay-up costs ($/year): administration, cost of capital and lay-up service and maintenance costs;
- Hire fixed costs ($/contract) and variable costs ($/day or $/hour): renting costs to cover administration, cost of capital and profit of the owner of the ship;
- Variable costs:
  - Daily running costs at ports ($/hours at port): port charges and fuel to maintain ships at port;
  - Fuel costs ($/nautical mile): fuel for ballast, parcel and full shipload sailing.

Although fuel cost is approximately proportional to the third power of the speed and, sometimes, even to the amount of cargo at the ship, this study considers that ships have different speeds and that the speed of each ship is fixed and given. This seems as a reasonable assumption for strategic and tactical planning problems.

Table 2 summarizes the options that a shipping company has for own and hire ships and the associated costs and revenues of each decision.

Table 2. Costs and revenues associated with each ship mix decision

Does the shipping company own the ship?	Was the ship used to serve voyages?	During the planning horizon, the ship was...	Associated Costs		Associated Revenues
Yes	Yes	**used**	•Fixed	•Variable	•Voyages
	No	**chartered**	•Fixed		•Charter fixed and variable
	Yes	**used and chartered**	•Fixed	•Variable	•Voyages •Charter fixed and variable
	No	**laid-up**	Laid-up		-
	No	**not used**	•None[a]		-
No	Yes	**hired**	•Hire fixed and variable		•Voyages
	No	**not used**	-		-

[a] *If the company has other fixed costs associated with an unused ship, these costs could be set as lay-up costs.*

3.3. Simplifying assumptions

This research encompasses most of real-life parameters, constraints and objectives. Still, some simplifying assumptions are considered to reduce combinatorial complexity. The assumptions below may be further addressed to bring problem solutions closer to reality.

a) Ships sail at a fixed speed; a ship could operate at lower speeds to reduce variable costs, but this is not taken into account here;

b) Ballast, full load and restricted waters speeds are the same for each ship during the whole voyage: in reality, these speeds are different and total sailing time is a composition of the number of nautical miles sailed on each speed;

c) Ships are available during the whole planning horizon with no interruption for maintenance: maintenance and out-of-service times are scheduled in advance for each ship and usually some ship will not be available during the whole planning horizon. This may be modeled as a "dummy" port call;

d) Inventory costs of goods in transit are not considered: each day that the cargo is onboard has a cost for the company that owns the cargo. Thus, the price of a voyage could be a function of the voyage duration.

4. Graph representation

In this section, we show the procedures that define contract and spot voyages as nodes of a directed graph with arcs denoting underlying sequences of voyages.

4.1. Sets

Consider the following sets: V as the set of ships, $VO \subset V$ as the set of owner's ships, $VHI=\{V\text{-}VO\}$ as the set of hired ships, $VLU \subset VO$ as the set of ships that may be laid-up,

$VCO=\{VO\text{-}VLU\}$ as the set of ships that may be chartered, P as the set of ports, CV as the set of contractual voyages, and SV as the set of spot voyages.

4.2. Ship parameters

A ship $v \in V$ has a net tonnage of NT_v, a sailing speed of SP_v knots, a fixed cost of FC_v US$/year, a cost of VC_v US$/nautical mile and a running cost at port $p \in P$ of RC_{vp} US$/hour. Loading and unloading time of ship $v \in V$ at port $p \in P$ is LUT_{vp} hours/ton. Each ship $v \in V$ is available for the first voyage of the planning horizon on time TA_v at port PA_v.

Each ship $v \in V$ has a waiting time in a queue at port $p \in P$ of Q_{vp} hours just before the start of service at each port call. If two or more subsequent port calls occur in the same port (e.g. the first port call is of a contractual voyage at port p and the second port call is of a spot voyage at the same port p), the ship waits a time Q_{vp} in a queue only once, before the first port call p. The distance in nautical miles between port $p \in P$ and $q \in P$ is DP_{pq}.

If a ship $vlu \in VLU$ is not used during the planning horizon, the shipping company pays a lay-up cost of LUC_{vlu} US$/year.

If a ship $vco \in VCO$ is chartered, the shipping company receives a fixed revenue of RCO_{vco} US$ for each time (up to twice during the planning horizon) that vco is chartered. Also, if $vco \in VCO$ is chartered, each chartered period must be within the time period $[BCO_{vco}, ECO_{vco}]$ in hours and the shipping company receives a variable income of $VRCO_{vco}$ US$ per chartered hour.

If a ship $vhi \in VHI$ is hired, the shipping company pays a fixed cost of FHI_{vhi} US$ and a cost of $VCHI_{vhi}$ US$/hour that vhi is hired. In addition, if $vhi \in VHI$ is hired, the hire period must be within the time period $[BHI_{vhi}, EHI_{vhi}]$ in hours.

4.3. Contractual and spot voyages parameters

After the queue waiting time at the first port call, service of a contractual voyage $i \in CV$ must start between the time interval $[BCV_i, ECV_i]$ and must not end after time TCV_i. The contractual voyage $i \in CV$ produces a revenue for the shipping company of RCV_i US$ if the company serves the whole set of $CVP_i \subseteq P$ port calls. The revenue RCV_i is net of any specific port fees associated with each port call. Service must follow the sequence order of ports of the set CVP_i. For a contractual voyage $i \in CV$, let $o_i(a)$, $a \in \{1,...,|CVP_i|\}$, represents the port call in position a of the ordered set CVP_i. At each port call $p \in CVP_i$, the ship must unload UCV_{ip} and load LCV_{ip} tons of cargo.

The service of a spot voyage $r \in SV$ must start, after the queue waiting time of the first port call, between the time interval $[BSV_r, ESV_r]$ and must not end after time TSV_r. The spot voyage $r \in SV$ produces a revenue for the shipping company of RSV_r US$ if the company loads LSV_r tons of cargo at port $LPSV_r$ and then unloads the same LSV_r tons of cargo at port $UPSV_r$. . The revenue RSV_r is net of any specific port fees associated with each port call of spot voyage $r \in SV$.

Let *latestTime* be the latest time of the planning horizon to serve all voyages, such as $latestTime = max(\{TCV_i : i \in CV\}, \{TSV_r : r \in SV\})$.

4.4. Node representation of voyages

The mathematical model is based on considering voyages as nodes of a graph. The arcs connecting the nodes represent the deployment of ships to voyages and also the sequencing of voyages within each ship route. Thus, before formulating the mathematical problem, the concept of representing voyages as nodes is presented.

Consider each contractual voyage and each port call of spot voyages as nodes of types 1 and 2, respectively. Figure 2 shows examples of the representation of nodes of types 1 and 2.

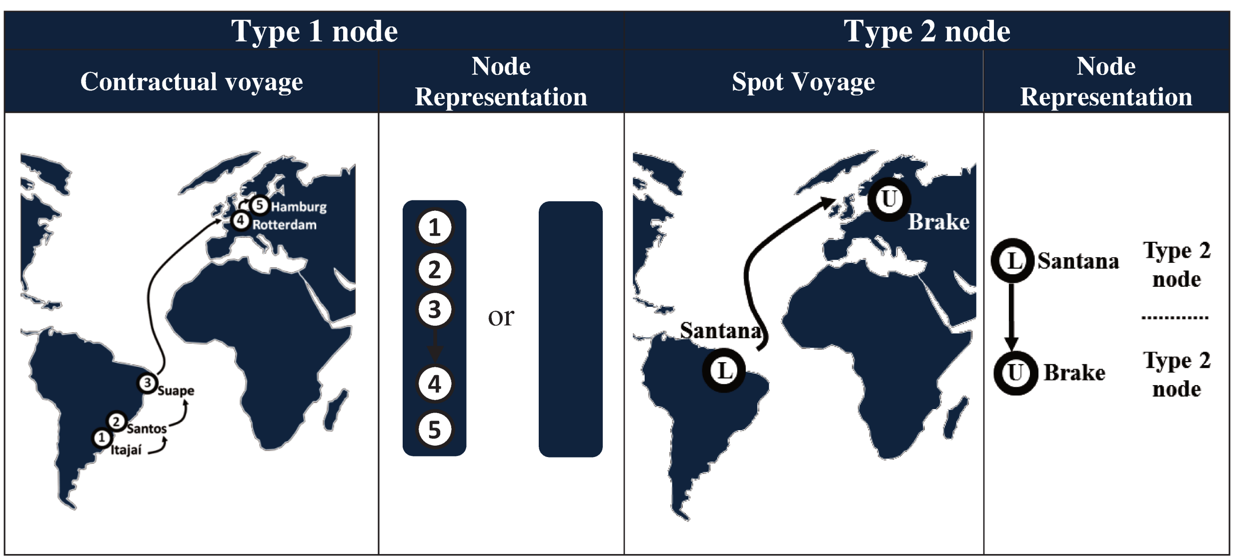


Figure 2. Example of node types 1 and 2

The left side of Figure 2 (type 1 node) shows how the contractual voyage Itajai→Santos→Suape→Rotterdam→Hamburg can be represented by a single type 1 node: a filled box with or without port calls numbers. The right side of Figure 2 (type 2 node) shows how the spot voyage Santana→Brake may be represented by two type 2 nodes. The first type 2 node is the loading port call at Santana, and the second type 2 node is the unloading port call at Brake.

Also, consider the combinations of contractual and spot voyages as nodes of type 3. Each type 3 node must have at least one port call of a spot voyage served within a contractual voyage. Figure 3 illustrates the combination of both nodes previously shown on Figure 2.

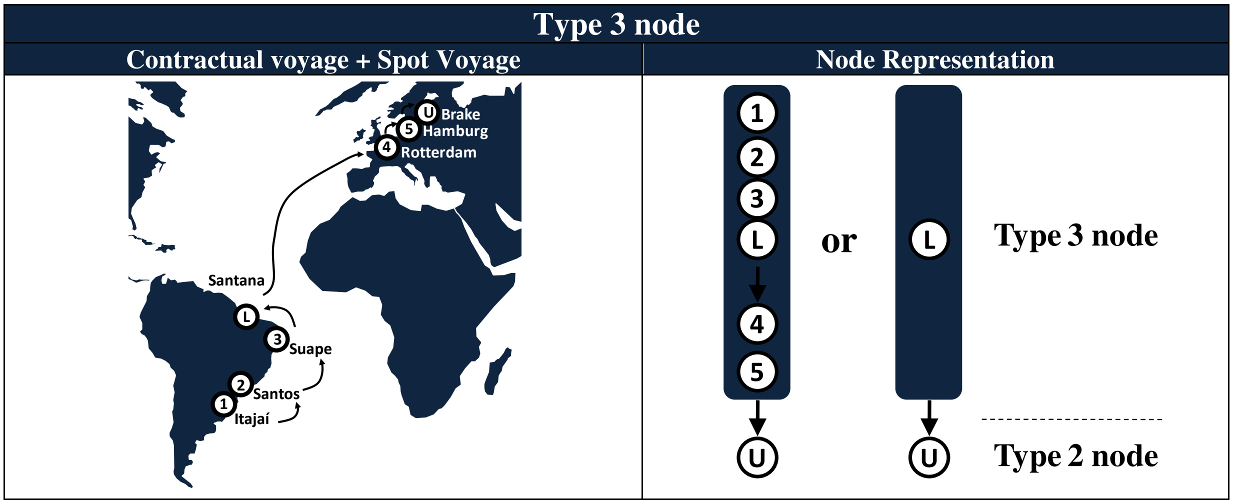


Figure 3. Example of a type 3 node

The resulting combination of the contractual and spot voyage is one type 3 node (Itajai→Santos→Suape→SANTANA→Rotterdam→Hamburg) and another type 2 node (unloading port call of the spot voyage at Brake). For simplicity purposes, the type 3 node representation on the right, which shows only the loading (L) or unloading (U) images of spot voyage port calls, is used by the authors for the rest of this document.

4.5. Sequence and combination of nodes of types 1, 2 and 3

The possible combinations of contractual and spot voyages depend on the positions of the port calls (loading and unloading) of the spot voyage between the first and last port calls of the contractual voyages. For instance, for a single spot voyage there are four possible positions: both port calls inside a contractual voyage, both outside, loading inside and unloading outside and vice-versa. Figure 4 shows the possible combinations of a single spot voyage to a number of contractual voyages between 1 and 3.

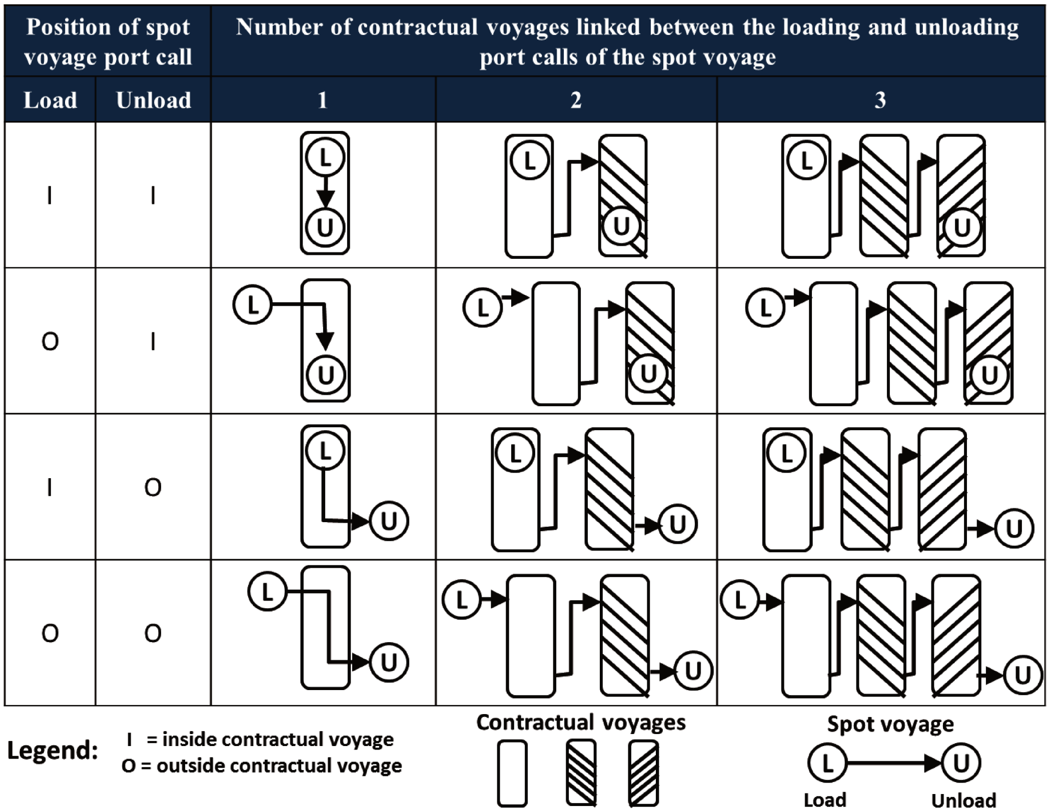


Figure 4. Example of combinations of contractual and spot voyages

A voyage node of type 3 is any contractual voyage that has at least one port call of a spot voyage served within the contractual voyage. Thus, any possible combination of spot and contractual voyages of Figure 4 may be represented through a sequence of voyage nodes of types 1, 2 and 3. Figure 5 shows how some sequences of contractual and spot voyages are represented through nodes.

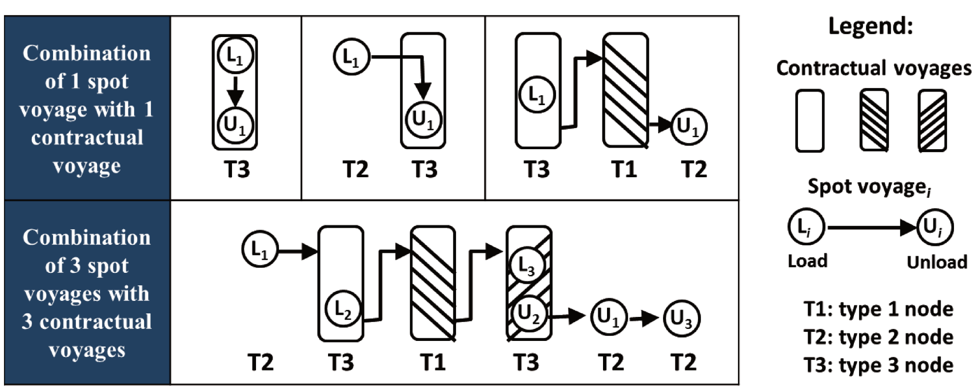


Figure 5. Node representation of combinations of contractual and spot voyages

In theory, there is no limit to the number of contractual and spot voyages that can be represented through a sequence of voyage nodes of types 1, 2 and 3. However, since a spot voyage $r \in SV$ usually has a time limit TSV_r between 15 and 90 days to be completed and the duration of a contractual voyage usually ranges from 7 to 45 days, it would be unusual, in practice, to combine a spot voyage to more than 3 contractual voyages.

If all contractual and spot voyages are represented through nodes, as well as the combinations of contractual and spot voyages, the mathematical problem may be formulated as the assignment of a sequence of nodes to each ship. A similar node model could be created for tramp shipping operations. For example, a type 1 node could contain a tramp shipping contractual voyage with a single port of origin and a single port of destination. Then, types 2 and 3 nodes could be constructed in a similar manner of Figure 5.

Since all contractual voyages must be served, the assignment of nodes to ships must guarantee that the selected set of type 1 and 3 nodes exactly covers all contractual voyages. Spot voyages, on the other hand, are optional. Thus, there are feasible solutions in which no nodes of types 2 and 3 are assigned to any ships.

4.6. Procedures to create the set of nodes and arcs

Three procedures construct the set of nodes of types 1, 2 and 3. This section presents the notation and the outcomes of these procedures. Refer to Appendix B for a detailed description on how each set of node type is created.

Let N be the set of voyage nodes and let $type_n \in \{1, 2, 3\}$ be the type of a voyage node $n \in N$. For a voyage node $n \in N$, let the set BC_n contain the indexes of the contractual voyages $i \in CV$ and the set BS_n contain the indexes of the spot voyages $r \in SV$ that are served by voyage node n.

Let the set $Ports_n$ contain the ordered sequence of port calls of voyage node n and let $o_n(a)$, $a \in \{1,...,|Ports_n|\}$, represents the port call in position a of the ordered set $Ports_n$.

Also, let $Demand_n$ be the net number of tons that node n loads (positive number) or unloads (negative number) to a ship and let $CapMin_n$ be the minimum idle capacity a ship must have to be able to serve node n.

For a ship $v \in V$ and voyage node $n \in N$, let $TotalTime_{vn}$ be the total time in hours spent by ship v during voyage node n and $Profit_{vn}$ be the total profit if ship v serves voyage node n. Finally, let $[BT_{vn}, ET_{vn}]$ be the time window to start service at voyage node n for ship $v \in V$.

As a result, the following node attributes are calculated:

- Total time spent by a ship in the node: queue, loading, unloading and traveling time between two consecutive ports;

- Acquired profit (or loss) if a ship serves the node;
- Earliest and latest time a ship may start service in the node to guarantee time window feasibility;
- Minimum requirement of idle capacity of ship just before service at the node.

In the procedure that creates the type 3 nodes, some feasible sequences of port calls are dominated by another sequence when considering $Profit_{vn}$, $CapMin_n$, $TotalTime_{vn}$ and $[BT_{vn}, ET_{vn}]$ values, which are the attributes of the nodes. In this case, the dominated feasible sequences are discarded. A sequence of a node a is dominated by a sequence of a node b if $Profit_a \leq Profit_b$ and $CapMin_a \geq CapMin_b$ and $TotalTime_a \geq TotalTime_b$ and $ET_a \leq ET_b$ and $BT_a + TotalTime_a \geq BT_b + TotalTime_b$, and there exists at least one strict inequality.

With the parameters of Sections 4.2 and 4.3 and the results of the calculations of the procedures, (see Appendix B) it is possible to present the mathematical formulation of the four models described in Table 1.

5. Mathematical formulation

Based on the nodes and parameters defined in the previous section, we present mixed integer programming formulations for the four variants of the FSMRTP shown in Table 1 (SIMPLE.Cv, SIMPLE.CvSv, FULL.Cv and FULL.CvSv).

5.1. Decision variables

Consider the following decision variables:

$\boldsymbol{z_v}$ = 1 if ship $v \in V$ is used during the planning horizon, 0 otherwise;

$\boldsymbol{xfirst_{vn}}$ = 1 if voyage node $n \in N$ is the first node of ship $v \in V$, 0 otherwise;

$\boldsymbol{xlast_{vn}}$ = 1 if voyage node $n \in N$ is the last node of ship $v \in V$, 0 otherwise;

$\boldsymbol{x_{vmn}}$ = 1 if ship $v \in V$ services voyage node $m \in N$ just before voyage node $n \in N$, 0 otherwise;

$\boldsymbol{st_{vn}} \in R_+$ = start time of ship $v \in V$ at voyage node $n \in N$;

$\boldsymbol{startTime_v} \in R_+$ = start time of the first port call of ship $v \in VCO$;

$\boldsymbol{endTime_v} \in R_+$ = end time of the last port call of ship $v \in VCO \cup VHI$;

$\boldsymbol{coph_v}$ = 1 if ship $v \in VCO$ is chartered for the whole planning horizon, 0 otherwise;

$\boldsymbol{cob_v}$ = 1 if ship $v \in VCO$ is chartered before $startTime_v$, 0 otherwise;

$\boldsymbol{coa_v}$ = 1 if ship $v \in VCO$ is chartered after $endTime_v$, 0 otherwise;

***cap**$_{vn}$* $\in R_+$ = available net tonnage capacity of ship $v \in V$ just before starting service at voyage node $n \in N;$

The mathematical formulation of each problem is presented next.

5.2. SIMPLE.Cv

The set N in the SIMPLE.Cv mathematical formulation contains only type 1 nodes and $VCO=VHI=VLU=\varnothing$.

$$
\begin{aligned}
Max\,total\;\; profit = & \sum_{v\in V}\sum_{m\in N}\sum_{n\in N, n\neq m} Profit_{vm}\left(x_{vmn}+xlast_{vm}\right)-\sum_{v\in V} FC_v z_v \\
& -\sum_{v\in V}\sum_{m\in N} VC_v DP_{PA_v o_m(1)} xfirst_{vm} - \sum_{v\in V}\sum_{m\in N}\sum_{n\in N, n\neq m} VC_v DP_{o_m(|Ports_m|)o_n(1)} x_{vmn} \\
& -\sum_{v\in V}\sum_{m\in N} RC_{vo_m(1)} Q_{vo_m(1)} xfirst_{vm} - \sum_{v\in V}\sum_{m\in N}\sum_{n\in N, n\neq m} RC_{vo_n(1)} Q_{vo_n(1)} x_{vmn}
\end{aligned}
$$

subject to:

$$\sum_{v\in V}\sum_{n\in N, n\neq m}\left(x_{vmn}+xlast_{vm}\right)=1, \qquad \{m\in N : i\in BC_m, i\in CV\} \qquad \textbf{(1)}$$

$$\sum_{m\in N} xfirst_{vm} = z_v, \qquad \forall v\in V \qquad \textbf{(2)}$$

$$xfirst_{vm} + \sum_{n\in N, n\neq m} x_{vnm} = \sum_{n\in N, n\neq m} x_{vmn} + xlast_{vm}, \qquad \forall v\in V, \forall m\in N \qquad \textbf{(3)}$$

$$st_{vm} \geq TA_v + \frac{DP_{PA_v o_m(1)}}{SP_v} + Q_{vo_m(1)} - M_1(1-xfirst_{vm}), \qquad \forall v\in V, \forall m\in N \qquad \textbf{(4)}$$

$$st_{vn} \geq st_{vm} + TotalTime_{vm} + \frac{DP_{o_m(|Ports_m|)o_n(1)}}{SP_v} + Q_{vo_n(1)} - M_2(1-x_{vmn}), \forall v\in V, \forall m,n\in N, m\neq n \qquad \textbf{(5)}$$

$$BT_{vm} \leq st_{vm} \leq ET_{vm}, \qquad \forall v\in V, \forall m\in N \qquad \textbf{(6)}$$

$$x_{vmn} \in \{0,1\}, \qquad \forall v\in V, \forall m,n\in N, m\neq n \qquad \textbf{(7)}$$

$$xfirst_{vm} \in \{0,1\}, \qquad \forall v\in V, \forall m\in N \qquad \textbf{(8)}$$

$$xlast_{vm} \in \{0,1\}, \qquad \forall v\in V, \forall m\in N \qquad \textbf{(9)}$$

$$z_v \in \{0,1\}, \qquad \forall v\in V \qquad \textbf{(10)}$$

The objective function maximizes total profit, which is the sum of each voyage node profit, minus the sum of fixed costs of the used ships, minus the sum of variable costs of the distance sailed before the first voyage node and between each pair of voyage nodes, minus the running costs associated to queue times of the first voyage node and between voyage nodes.

Constraints (1) guarantee that every contractual voyage is served exactly once. A contractual voyage i is served if a node m that contains this contractual voyage ($i \in BC_m$) is visited by a ship v. In a sequence of nodes of a ship v, node m is either before some other node n (x_{vmn}=1) or node m is the last node of the route ($xlast_{vm}$=1). Thus, if node m is visited by ship v, the expression $\sum_{n \in N, n \neq m} \left(x_{vmn} + xlast_{vm} \right)$ must be equal to 1. There are mathematical formulations, such as those in Christiansen et al. (2007), which use artificial origin and destination nodes to avoid the notation of the variables *xfirst* and *xlast*. Either approach is valid since both models have the same total number of variables.

Constraints (2) determine that if a ship v is used, ship v must exactly have one first node m ($xfirst_{vm}$=1). Constraints (3) are the flow constraints. These constraints balance the input and output flow of every voyage node. Note that the right hand side of the constraints ($\sum_{n \in N, n \neq m} x_{vmn} + xlast_{vm}$) is at most equal to 1 because of constraints (1). Thus, the left hand side ($xfirst_{vm} + \sum_{n \in N, n \neq m} x_{vnm}$) is also either equal to zero or to 1.

Constraints (4) and (5) calculate the start time of each voyage node for each ship. Constraints (4) calculate the start time for the first node and constraints (5) for the subsequent nodes. In addition, time window constraints for each voyage node are represented by (6).

M_1 and M_2 are large numbers that assure that if either $xfirst_{vm}$ or x_{vmn} is zero, the right hand side of constraints (4) and (5) will be at most BT_{vm} and BT_{vn} respectively. Appendix C presents additional information on the calculation of all large numbers of the mathematical models.

The SIMPLE.Cv and FULL.Cv models do not require capacity constraints because $Demand_n = 0, \forall n \in N$, since all cargo loaded on a type 1 node is unloaded within the same node. Constraints (7) to (10) define the domain of the variables.

5.3. SIMPLE.CvSv

The set N in the SIMPLE.CvSv mathematical formulation contains nodes of types 1, 2 and 3 and $VCO=VHI=VLU=\varnothing$. The SIMPLE.CvSv model consists of the objective function and all of the constraints of the SIMPLE.Cv model with the addition of following spot voyages constraints (11), (12), (13) and capacity constraints (14), (15).

$$\sum_{v \in V} \sum_{n \in N, n \neq m} \left(x_{vmn} + xlast_{vm} \right) \leq 1, \qquad \{ m \in N : UPSV_r \in Ports_m, r \in SV \} \tag{11}$$

$$\sum_{\{m \in N : LPSV_r \in Ports_m, r \in SV\}} \sum_{n \in N, n \neq m} \left(xfirst_{vm} + x_{vnm} \right) = \sum_{\{h \in N : UPSV_r \in Ports_h, r \in SV\}} \sum_{n \in N, n \neq h} \left(x_{vhn} + xlast_{vh} \right), \qquad \forall v \tag{12}$$

$$st_{vn} \geq st_{vm} + TotalTime_{vm} + \frac{DP_{o_m(|Ports_m|)o_n(1)}}{SP_v} + Q_{vo_n(1)} - M_2(1 - \sum_{h \in N, h \neq m} x_{vmh}), \forall v \in V$$
$$\{ m \in N : LPSV_r \in Ports_m, UPSV_r \notin Ports_m, r \in SV \}, \{ n \in N : LPSV_r \notin Ports_m, UPSV_r \in Ports_n, r \in SV \} \tag{13}$$

$$cap_{vn} \leq cap_{vm} - Demand_m + M_3(1 - x_{vmn}), \qquad \forall v \in V, \forall m, n \in N, m \neq n \tag{14}$$

$$CapMin_m \leq cap_{vm} \leq NT_v, \qquad \forall v \in V, \forall m \in N \tag{15}$$

Constraints (11) ensure that spot voyages are served at most once. A spot voyage r may be served either through a type 2 unloading node m (in which $Ports_m = UPSV_r$) or through a type 3 node n that contains the unloading port call of r ($UPSV_r \in Ports_n$). Therefore, constraints (11) guarantee that the unloading port call of spot voyage r may be served at most once by a ship v.

If the unloading port call of spot voyage r is served by ship v, constraints (12) ensure that the correspondent loading port call of spot voyage r must also be served exactly once by the same ship v. Thus, constraints (11) and (12) together guarantee that if a cargo of a spot voyage is loaded, then this cargo must be unloaded by the same ship.

Constraints (13) ensure that the start time of the unload port call of spot voyage r is greater than or equal to the earliest time that it would be possible to start unloading the cargo of spot voyage r. For example, consider the infeasible sequence of three nodes [U_1]-[L_2]-[C_3-L_1], where L_1 and U_1 are, respectively, the loading and unloading port calls of spot voyage 1, L_2 is the loading port call of spot voyage 2, and C_3 is the first port call of contractual voyage 3. This sequence might occur if the time window of a spot voyage 1 is too wide, however, this case did not arise in the computational experiments.

Net tonnage capacity constraints are represented by (14). These constraints calculate the ship available capacity to serve node n just after node m if x_{vmn} is equal to 1. Constraints (15) establish that every ship must have enough idle capacity to serve a node.

5.4. FULL.Cv

The set N in the FULL.Cv mathematical formulation contains only type 1 nodes. In this model there are ships that may be laid-up, chartered and/or hired. Also, let the variables

$cob_v, coa_v, coph_v \in \{0,1\}$ determine, respectively, whether ship $v \in V$ is chartered before, after or for the whole planning horizon.

$$
\begin{aligned}
Max\ total\ profit = & \sum_{v\in V}\sum_{m\in N}\sum_{n\in N, n\neq m} Profit_{vm}\left(x_{vmn} + xlast_{vm}\right) - \sum_{v\in VO} FC_v z_v - \sum_{v\in VLU} LUC_v(1-z_v) \\
& - \sum_{v\in VCO} FC_v coph_v + \sum_{v\in VCO} RCO_v\left(cob_v + coa_v\right) + \sum_{v\in VCO} VRCO_v\left(startTime_v - TA_v\right) \\
& + \sum_{v\in VCO} VRCO_v\left(latestTime - endTime_v\right) \\
& - \sum_{v\in VHI} FHI_v z_v - \sum_{v\in VHI} VCHI_v\left(endTime_v - TA_v\right) \\
& - \sum_{v\in V}\sum_{m\in N} VC_v DP_{PA_v o_m(1)} xfirst_{vm} - \sum_{v\in V}\sum_{m\in N}\sum_{n\in N, n\neq m} VC_v DP_{o_m(|Ports_m|)o_n(1)} x_{vmn} \\
& - \sum_{v\in V}\sum_{m\in N} RC_{vo_m(1)} Q_{vo_m(1)} xfirst_{vm} - \sum_{v\in V}\sum_{m\in N}\sum_{n\in N, n\neq m} RC_{vo_n(1)} Q_{vo_n(1)} x_{vmn}
\end{aligned}
$$

subject to constraints (1) to (6) of SIMPLE.Cv and:

$$startTime_v \leq st_{vm} + M_4(1 - xfirst_{vm}), \quad \forall v \in VCO, \forall m \in N \quad \textbf{(16)}$$

$$startTime_v - TA_v \geq BCO_v cob_v, \quad \forall v \in VCO \quad \textbf{(17)}$$

$$startTime_v - TA_v \leq ECO_v cob_v, \quad \forall v \in VCO \quad \textbf{(18)}$$

$$endTime_v \geq st_{vm} + TotalTime_{vm} - M_5(1 - xlast_{vm}), \quad \forall v \in \{VCO \cup VHI\}, \forall m \in N \quad \textbf{(19)}$$

$$latestTime - endTime_v \geq BCO_v coa_v, \quad \forall v \in VCO \quad \textbf{(20)}$$

$$latestTime - endTime_v \leq ECO_v coa_v, \quad \forall v \in VCO \quad \textbf{(21)}$$

$$endTime_v \geq startTime_v, \quad \forall v \in VCO \quad \textbf{(22)}$$

$$endTime_v - TA_v \geq BHI_v z_v, \quad \forall v \in VHI \quad \textbf{(23)}$$

$$endTime_v - TA_v \leq EHI_v z_v, \quad \forall v \in VHI \quad \textbf{(24)}$$

$$TA_v \leq startTime_v \leq latestTime, \quad \forall v \in VCO \quad \textbf{(25)}$$

$$TA_v \leq endTime_v \leq latestTime, \quad \forall v \in \{VCO \cup VHI\} \quad \textbf{(26)}$$

$$coph_v \geq cob_v - z_v, \quad \forall v \in \{VCO\} \quad \textbf{(27)}$$

$$coph_v \geq coa_v - z_v, \quad \forall v \in \{VCO\} \quad \textbf{(28)}$$

$$x_{vmn} \in \{0,1\}, \quad \forall v \in V, \forall m,n \in N, m \neq n \quad \textbf{(29)}$$

$$xfirst_{vm} \in \{0,1\}, \quad \forall v \in V, \forall m \in N \quad \textbf{(30)}$$

$$xlast_{vm} \in \{0,1\}, \qquad \forall v \in V, \forall m \in N \qquad \textbf{(31)}$$

$$z_v \in \{0,1\}, \qquad \forall v \in V \qquad \textbf{(32)}$$

$$cob_v \in \{0,1\}, \qquad \forall v \in V \qquad \textbf{(33)}$$

$$coa_v \in \{0,1\}, \qquad \forall v \in V \qquad \textbf{(34)}$$

$$coph_v \in \{0,1\}, \qquad \forall v \in V \qquad \textbf{(35)}$$

The objective function of FULL.Cv has additional elements when compared with the objective function of SIMPLE.Cv. Each additional expression is explained in Table 3.

Table 3. Additional expressions present on the FULL.Cv objective function

Additional element of the FULL.Cv objective function	Description
$\sum_{v \in VLU} LUC_v (1 - z_v)$	Fixed lay-up cost if the ship v is not used during the planning horizon.
$\sum_{v \in VCO} FC_v coph_v$	Fixed cost if the ship v is chartered for the whole planning horizon. This cost is the same as in the situation in which the ship is used to serve voyages.
$\sum_{v \in VCO} RCO_v (cob_v + coa_v)$	Fixed charter revenue if the ship v is chartered before (cob_v=1) and/or after (coa_v=1) the period of time that the ship was used. If the ship is not used, it may be also chartered up to two times (cob_v=coa_v=1).
$\sum_{v \in VCO} VRCO_v (startTime_v - TA_v)$	Variable charter revenue if the ship v is chartered before (cob_v must be equal to 1). If the ship v is not chartered before (cob_v=0), constraints (17) and (18) guarantee that $startTime_v = TA_v$.
$\sum_{v \in VCO} VRCO_v (latestTime - endTime_v)$	Variable charter revenue if the ship v is chartered after (coa_v must be equal to 1). If the ship v is not chartered after (coa_v=0), constraints (20) and (21) guarantee that $endTime_v = latestTime$.
$\sum_{v \in VHI} FHI_v z_v$	Fixed cost of hiring the ship v.
$\sum_{v \in VHI} VCHI_v (endTime_v - TA_v)$	Variable cost of hiring the ship v. The cost is proportional to the period of time that the ship was used. If the ship v was not hired (z_v=0), constraints (23) and (24) guarantee that $endTime_v = TA_v$.

Constraints (16) establish the limit for each variable $startTime_v$ of ship $v \in VCO$ to be the start time of the very first port call of ship v at node m (when $xfirst_{vm}$=1). M_4 is a large number that guarantees that $startTime_v$ is only limited if $xfirst_{vm}$=1.

Constraints (17) and (18) guarantee that the amount of chartered time before is within the time interval [BCO_v, ECO_v]. If the ship is not chartered before, $startTime_v = TA_v$, resulting in no variable charter revenue in the objective function.

Constraints (19) establish the limit for each variable $endTime_v$ of ship $v \in VCO \cup VHI$ to be the end time of the last port call of ship v at node m (when $xlast_{vm}$=1). M_5 is a large number that guarantees that $endTime_v$ is only limited if $xlast_{vm}$=1.

Constraints (20) and (21) ensure that the amount of chartered time after is within the time interval [BCO_v, ECO_v]. If the ship is not chartered after, $endTime_v = latestTime$, resulting in no variable charter revenue in the objective function.

Constraints (22) assure that $endTime_v$ is always greater than or equal to $startTime_v$ $\forall v \in VCO$. Although constraints (22) may seem redundant, they are necessary when a ship $\forall v \in VCO$ is chartered both before and after ($cob_v = coa_v = 1$) and ship v is not used to serve any voyages (z_v=0). In this case, $startTime_v$ and $endTime_v$ are bounded by (17), (18), (20) and (21) and, depending of the values of TA_v, BCO_v and ECO_v, $endTime_v$ could assume values lower than $startTime_v$ (for an example of this situation, see Appendix C).

Constraints (23) and (24) guarantee that the amount of hired in time is within the time interval [BHI_v, EHI_v]. If the ship is not hired, $endTime_v = TA_v$, resulting in no variable hire cost in the objective function.

Constraints (25) and (26) impose the bounds of $startTime_v$ and $endTime_v$ to the time interval [TA_v, $latestTime$].

Constraints (27) and (28) ensure that $coph_v$=1 if z_v=0 and either cob_v=1 or coa_v=1 or both cob_v=coa_v=1. These constraints guarantee that the fixed cost of ship $v \in VCO$ is subtracted in the objective function in the case that z_v=0 and the ship v is used to be chartered at least once. Finally, constraints (33) to (35) define the domain for the charter variables.

5.5. FULL.CvSv

The set N in the FULL.CvSv mathematical formulation contains nodes of types 1, 2 and 3. As in the SIMPLE.CvSv mathematical model, the variant FULL.CvSv requires the additional spot voyages and capacity constraints (11-15).

6. Computational experiments

Computational experiments were executed with two solution approaches, MIP exact (MIPE) and MIP best nodes (MIPBN). MIPE constructs a graph based on the three types of voyage nodes and then creates the mixed integer programming (MIP) model. Subsequently, the MIP model is solved with the commercial solver software CPLEX. MIPBN is similar to the MIPE. However, the graph of MIPBN contains only a selected number of best (top) nodes. MIPBN does not guarantee the existence of an optimal solution since not all nodes are inserted into the graph. The smaller number of nodes in the graph implies that the mathematical model has fewer variables and constraints and may be solved by CPLEX in a shorter computational time.

Figure 6 describes the MIPBN solution method and how best nodes are selected using two parameters, top_1 and top_2. Top_1 represents the number of best type 3 nodes to select in each triplet "contractual voyage x subset of spot voyages x ship" and top_2 the number of best type 3 nodes to select in each pair "contractual voyage x ship".

1. Create the nodes of types 1, 2 and 3.
2. Keep only the best top_1 type 3 nodes of the set N:
    2.1. For each contractual voyage $i \in CV$ and each subset $S \in SV$ of spot voyages, do:
        2.1.1. Create the set $ST3 = \{n \in N : type_n = 3, i = BC_n, S = BS_n\}$
        2.1.2. Create the sets $ST3_v \subseteq ST3$ for each ship $v \in V$ that may serve the nodes of set $ST3$.
        2.1.3. Sort the elements of set $ST3_v$ in $list_1$ considering two hierarchical criteria: (i) the decreasing order of total $EstimatedProfit_{vn} = Profit_{vn}$ + revenues and costs associated with the loading (unloading) spot voyages port calls just before (after) node n, (ii) the increasing order of $TotalTime_{vn}$.
        2.1.4. Remove from the set N the nodes $m \in ST3_v$ that are not in the top_1 nodes of $list_1$.
3. Keep only the best top_2 type 3 nodes:
    3.1. For each contractual voyage $i \in CV$, do:
        3.1.1. Create the set $ST3_v$ of type 3 nodes n in which $i = BC_n$ and node n may be served by ship $v \in V$.
        3.1.2. Sort the elements of the set $ST3_v$ in $list_2$ considering the same two hierarchical criteria of step 2.1.3.
        3.1.3. Remove from the set N the nodes $m \in ST3_v$ that are not in the top_2 nodes of $list_2$.
4. Create a graph with the set of feasible arcs connecting the nodes of all types
5. Create the MIP model (variables, objective function and constraints)
6. Solve the MIP model using CPLEX
7. Return the best feasible solution found by CPLEX

Figure 6. MIPBN solution method

The same set of spot voyages may be combined with a single contractual voyage in different manners depending on the positioning of the spot voyage port calls within the contractual voyage. Therefore, for every triplet "contractual voyage x subsets of spot voyages x ship", step 2 of Figure 6 creates a set of type 3 nodes with different sequences of port calls and stores this set of nodes in the ordered list $list_1$. To sort the nodes of $list_1$, a node A is considered to be better than another node B, when both nodes are served by the same ship $v \in V$, if $EstimatedProfit_{vA} > EstimatedProfit_{vB}$, or, if $EstimatedProfit_{vA} = EstimatedProfit_{vB}$ and $TotalTime_{vA} < TotalTime_{vB}$. For any type 3 node $n \in N$, $EstimatedProfit_{vn}$ is defined as the total profit if ship $v \in V$ serves node $n \in N$ and also serves all the other required spot voyage port calls to ensure to ensure route feasibility, either loading or unloading port calls. For example, suppose that node n contains the loading port call of a spot voyage $r \in SV$. A route with node n would only be feasible if the unloading port call of spot voyage r is served sometime after node n. Since there may be more than one route in which the unloading port call of spot voyage r is served after node n, $EstimatedProfit_{vn}$ is an estimate in which the unloading port call of spot voyage r occurs immediately after node n. The second comparison $TotalTime_{vA} < TotalTime_{vB}$ ranks first nodes that consume less time, and consequently fewer resources, of ship v to serve the same set of contractual and spot voyages.

After all feasible combinations have been implemented and filtered by step 2 considering the triplet "contractual voyage *x* subsets of spot voyages *x* ship", step 3 sorts out the best nodes, using the same criteria of the previous step, for every pair "contractual voyage *x* ship". In other words, at the end of step 3 only the top_2 type 3 best nodes of every pair "contractual voyage *x* ship" are kept in set N. Steps 4, 5, 6 and 7 creates the graph, the MIP model and solve the problem with CPLEX.

In the computational study, two variations of MIPBN were tested. For the first variant, MIPBN00, the parameters top_1 and top_2 were set to zero which means that no type 3 nodes are inserted into the graph. The second variant, MIPBN15, the parameters top_1 and top_2 were set, respectively, to one and five. Larger values of top_1 and top_2 were also tested, for example, top_1=3 and top_2=10. However, in this case the number of nodes was still large, almost the same number of nodes of MIPE, which resulted in an experiment, regarding both solution quality and computational time, similar to the one of MIPE.

6.1. Set of test problems

A random test problem generator that considers real world assumptions and parameters was developed to create a set of 56 test problems (14 test problems for each one of the 4 models of Table 1). Although the test problems do not reflect a particular liner operation, the parameters of each problem were defined within a range of values that could represent practical ship routing operations.

The number of ships of a company ranges from 18 to 50 ships plus 6 to 32 hire ships, resulting in a total of 24 to 82 ships. Each ship belongs to a ship class, such as Handysize (two types of Handysize), Supramax, Panamax and Capesize (also two types of Capesize), which determines most of ships parameters such as capacity, sailing speed and costs. Ships are available for the planning horizon at different times because each ship must conclude current service before it receives a new route assignment. The time at which a ship becomes available for planning is chosen randomly between 0 and 20 days and the time of availability of hire ships is chosen between 0 and 150 days.

A set of 53 worldwide ports has been used to create the set of demands, and their latitude and longitude define a distance matrix among ports. In addition, for each pair 'port *x* ship class', a service time, loading/unloading rate and running costs were randomly determined within real world assumptions. Also, the location of the ship at the time the ship is available is a port randomly selected from the set of 53 ports.

It may not be possible for a certain port to serve all types of ship class. For instance, a port may not be able to accommodate a large ship or the port may not have the adequate loading/unloading equipment. To simulate this constraint, the loading and unloading time LUT_{vp} is set to a maximum limit for some 'port *x* ship class' pairs (approximately 2% of all 53 x 6 =318 pairs).

Each problem of the data set has between 30 and 110 contractual voyages and between 10 and 34 spot voyages (ratio of contractual and spot voyages around 75%). The amount of cargo at each port call, the duration and time windows of voyages, and revenue are selected randomly. In addition, contractual voyages were constructed based on a set of 12 trade routes with a number of port calls between 15 and 30.

Spot voyages have a single port call to load and a single port call to unload the cargo. The origin and destination port calls are chosen randomly from the set of 53 ports.

A voyage may not be served by all classes of ships (e.g. product incompatibility or commercial constraints). To simulate this constraint, there is a uniform probability of 4% of each ship class not serving each contractual or spot voyage. At last, the earliest time to start each voyage is chosen randomly so that the latest time is within the planning horizon.

6.2. Computational results

The graph and MIP models were implemented in C++ and computational tests were executed in a Intel Xeon 2.83 GHz, 8Gb RAM computer with Ubuntu operating system. CPLEX 12.4 was used to solve the MIP problems with the optimization time limit set at 24 hours. The summary of the results of the computational experiments is shown below (see Appendix D for the complete results):

- MIPE solved 100% of the 14 SIMPLE.Cv test problems to optimality within a maximum computational time of 17 minutes;

- MIPE also solved eleven FULL.Cv test problems (79%) to optimality within a maximum computational time of 12 hours. The maximum optimality gap among the three test problems that were not solved to optimality by MIPE is 4.2%;
- Six SIMPLE.CvSv test problems (43%) were solved to optimality by MIPE. Most of the remaining problems were best solved by MIPBN15;
- MIPE efficiently solved only the three smallest FULL.CvSv test problems. Although MIPBN00 performed best for most of the other FULL.CvSv test problems, the quality of the feasible solutions found by MIPBN00 for the largest FULL.CvSv test problems are inferior to that of the best solutions found for the respective test problems without spot voyages. In addition, no solution method was able to find a feasible solution for the largest FULL.CvSv test problem *n*. Thus, the proposed MIP models proved to be inappropriate to solve large FULL.CvSv problems (e.g. fleet between 40 and 80 ships);
- The profit obtained by the proposed MIP models in the FULL problems is always (at least 50%) better than those obtained in the SIMPLE problems. The greater profit of the FULL models derives from the additional charter revenues and also from the option of using cheaper hire ships.

7. Conclusions

This research presented a generic mixed integer mathematical programming model to tackle planning problems faced by liner shipping companies in maritime logistics. Computational tests were executed on a proposed set of 56 test problems that were based on real world data. Test results suggest that exact methods were able to solve small to medium problems. However, the ability to obtain feasible and high quality solutions with such methods is reduced as problem size increases. In this case, models based on incomplete and smaller graphs showed more adequate.

Although the computational time required by the MIPE and MIPBN methods are high for some medium sized test problems, the optimization process could be stopped after about 5 to 8 hours of CPU time if there is no need to find the optimal solution nor to prove that the incumbent solution is optimal. For larger problems, even the time limit of 24 hours was not enough to close the gap between the best node and the incumbent solution. Therefore, heuristic approaches, such as tabu search, may be the best option in situations in which only a short period of computational time is available.

Finally, the following topics of future work related to this research are presented.

- Implement rounding and local search heuristics to find feasible solutions earlier in the branch-and-bound tree. These heuristics could support CPLEX to reduce overall computational time.
- Investigate new policies for the selection of the next variable for branching (and to which branching direction). For example, the profit estimate information could be used to increase the priority of variables related to high profitable contractual and spot voyages after a feasible solution has been found.
- Include transshipment into the MIP model. Transshipment is the possibility of moving a cargo from one ship to another at an intermediate port and could be included in the model in an approximately manner. For example, extra type 3 nodes, each one with a complete or partial route of feasible transshipments, could be added. It may be impossible to add all feasible transshipment operations because of the large number of combinations, but it may be reasonable to add the most promising ones.
- Use other types of nodes to include to the model additional maritime operations, such as refueling, maintenance or extra charter contracts, in a similar manner to the modeling of contractual and spot voyages. These operations need to be ship specific and to occur at pre-determined ports. Also, the operations may be mandatory or optional, may have service time windows and may produce either revenues or costs.
- Test the proposed solution methods on problems that have many voyages with a small number (e.g. one) of pickup and deliver port calls. These types of voyages, common to tramp operations, may further increase the combinatorial complexity because of the larger number of feasible routes.

References

Agarwal, R., & Ergun, Ö. (2008). Ship scheduling and network design for cargo routing in liner shipping. *Transportation Science*, 42(2), 175–196.

Besbes, O., & Savin, S. (2009). Going bunkers: the joint route selection and refueling problem. *Manufacturing & Service Operations Management*, 11(4), 694-711.

Cho, S. C., & Perakis, A. N. (1996). Optimal liner fleet routeing strategies. *Maritime Policy & Management*, 23(3), 249-259.

Christiansen, M., Fagerholt, K., & Ronen, D. (2004). Ship routing and scheduling – status and perspectives. *Transportation Science*, 38(1), 1-18.

Christiansen, M., Fagerholt, K., Nygreen, B., & Ronen, D. (2007). Handbooks in Operations Research and Management Science: Transportation. In C. Barnhart, & G. Laporte (Eds), *Maritime Transportation* (pp. 189-284). Amsterdam: North-Holland.

Christiansen, M., Fagerholt, K., Hasle, G., Nygreen, B., & Minsaas, A. (2009). Maritime transport optimization: An ocean of opportunities. *OR&MS Today*, 36(2), 26-31.

Christiansen, M., Fagerholt, K., Nygreen, B., & Ronen, D. (2012). Ship Routing and Scheduling in the New Millennium. *European Journal of Operational Research*, in press, doi: http://dx.doi.org/10.1016/j.ejor.2012.12.002.

Fagerholt, K. (2001). Ship scheduling with soft time windows – an optimization based approach. *European Journal of Operational Research*, 131(3), 559-571.

Fagerholt, K. (2004). A computer-based decision support system for ship fleet scheduling – Experience and future research. *Decision Support Systems*, 37(1), 35-47.

Fagerholt, K., Christiansen, M., Hvattum, L. M., Johnsen, T. A. V., & Vabø, T. J. (2010). A decision support methodology for strategic planning in maritime transportation. *OMEGA – The International of Management Science*, 38, 465-474.

Gelareh, S., & Meng, Q. (2010). A novel modeling approach for the fleet deployment problem within a short-term planning horizon. *Transportation Research Part E*, 46,76-89.

Hennig, F., Nygreen, B., Christiansen, M., Fagerholt, K., Furman, K. C., Song, J., Kocis, G. R., & Warrick, P. H. (2012). Maritime crude oil transportation – A split pickup and split delivery problem. *European Journal of Operational Research*, 218, 764-774.

Kim, S-H., & Lee, K-K. (1997). An optimization-based decision support system for ship scheduling. *Computers Industrial Engineering*, 33(3-4), 689-692.

Lawrence, S. A. (1972). International sea transport: the years ahead. Lexington Books, Lexington, MA.

Meng, Q., Wang, T., & Wang, S. (2012). Short-term liner ship fleet planning with container transshipment and uncertain container shipment demand. *European Journal of Operational Research*, 223, 96-105.

Psarafatis, H. N., & Kontovas, C. A. (2013). Speed models for energy-efficient maritime transportation: A taxonomy and survey. *Transportation Research Part C*, 26, 331-351.

Ronen, D. (1983). Cargo ships routing and scheduling: Survey of models and problem. *European Journal of Operational Research*, 12, 119-126.

Ronen, D. (1993). Ship scheduling: The last decade. *European Journal of Operational Research*, 71(3), 325-333.

UNCTAD. (2011). Review of maritime transport. United Nations, New York and Geneva.

Yan, S., Chen, C-Y., & Lin, S-C. (2009) Ship scheduling and container shipment planning for liners in short-term operations. *Journal of Maritime Science Technology*, 14, 417-435.

Appendix A – Additional information on liner, tramp and industrial operations

Table A.1. Description of the liner, tramp and industrial operations

	Liner	Tramp	Industrial
Business model	•The company establishes trade routes and frequency of service (similar to bus itineraries) •The regional offices sell available capacity to shippers, usually yearly contracts, with a few months in advance. These contracts, once accepted, are mandatory to be served and may represent from 80% to 95% of the liner company's business •The company attempts to fill idleness with spot demand	•The company (usually) owns a small or a medium fleet of ships and sells its capacity in two-ways: ‣ Contracts of affreightment: commitment to move a certain amount of cargo from Port 'A' to Port 'B' within a determined time window. May require one or more voyages and could be full or parcel shipload ‣ Spot voyages: commitment to serve a voyage from Port 'A' to Port 'B' (similar to a taxi cab, usually full shipload)	•An industrial company has a high volume of liquid or dry bulk cargo to transport, such as oil, chemicals, minerals or grains •Because of the high volume, the company decides to own or control a fleet of ships, including decisions of ship design, number of ships, routes and scheduling •The company often faces supply chain decisions, considering the integrated planning of production, inventory and port operations
Cargo owner	Shipper (client of the liner company)	Shipper (client of the tramp company)	Industrial company
Ship owner	•Liner company •May hire other ships (time charter contract) if profitable or if necessary to fulfill mandatory contracts •May charter, lay-up or scrap excess ships	•Tramp company •May hire other ships (time charter contract) if profitable to accept contracts of affreightment or to accommodate a peak of spot voyages demand •May charter, lay-up or scrap excess ships	•Industrial company •If possible, the company sizes the number of ships below its long term needs and hire other ships (time charter contract from tramp companies) to accommodate peak demand
Financial objective	To maximize profit	To maximize profit	To minimize overall supply chain and transportation costs of moving all cargo within the planning horizon
Key decisions	•Trade routes and itinerary •Transshipment of containers •Positioning of empty containers •Fleet mix and deployment •Scheduling of spot voyages, which are usually accepted if there is enough capacity •Balanced ship loading	•Fleet mix and deployment •Ship routing and scheduling •Mix between contracts of affreightment and spot voyages (decision mainly depends on the estimate of the future price of the spot voyage market)	•Fleet mix and deployment •Ship scheduling •Integrated supply chain planning •Inventory and environmental routing to fulfill the supply chain plan (e.g. avoid product stock-out)
Type of ships and cargo	•Containers •General cargo (e.g. packaged goods in ship's hold or deck) •Refrigerated loads •Roll-on-Rool-off (Ro-Ro ramps for trucks/cars)	•Dry bulk (e.g. grain, coal, iron ore and other minerals) •Tankers (crude oil, liquids in bulk) •Liquefied gas •Refrigerated ships	•Dry bulk •Tankers •Liquefied gas •Specialized cargo with specific packaging and/or handling equipment
Size and frequency of shipments	•Parcel shiploads: large number of port calls and clients per voyage (e.g. container shipment) •Routes may be served weekly (or even daily) and through more than one ship (e.g. a traderoute with a 4 week duration and a weekly frequency requires at least 4 ships)	•Full shipload: a pick-up and delivery from a single port of origin to a single port of destination •Parcel shipload: few clients and port calls per voyage •Frequency depends on the volume of the contracts of affreightment	•Full shipload (usually): a pick-up and delivery from a single port of origin to a single port of destination •Frequency depends of the integrated supply chain planning

The tactical problems approached by this study are outlined by the gray shaded cells of Table A.2.

Table A.2. Summary of maritime problems and relative importance

	Example of problems	Liner	Tramp	Industrial
Strategic	•Market and trade selection ▸ Which markets and geographies should be serviced?	+++	+++	++
	•Ship design ▸ What is the optimal size of the ship? ▸ What on board loading/unloading equipment are needed?	++	++	+++
	•Network and transportation system design ▸ Are hubs and transshipment ports desirable? ▸ Are there intermodal (rail, road, barge) integrated services? •Fixed route/itinerary determination (e.g. trade routes) ▸ What are the port calls and frequency of service of each trade route?	+++	-	++
	•Fleet size and mix decisions ▸ How many ships, and of which type, should be on the fleet? ▸ Should excess ships be scrapped?	+++	+++	+++
	•Contract evaluation ▸ Which long term contracts should be taken? ▸ How can a company be hedged against spot market price change?	++	+++	-
	•Port/terminal location, size and design •Supply chain planning ▸ How is shipping affected by, and how it affects, production, inventory and other integrated processes of the supply chain?	-	-	+++
Tactical	•Fleet size and mix decisions ▸ Are changes on current fleet size/mix desirable? ▸ Which hired and/or chartered contracts should be engaged?	+++	+++	+++
	•Contract evaluation ▸ Which short term (spot) contracts should be taken?	++	+++	-
	•Fleet deployment ▸ How many and which ships should serve which trade routes?	+++	-	-
	•Ship routing ▸ What is the best sequence of port calls for each ship?	++	+++	+
	•Ship scheduling ▸ When should each ship start/end service at each port call?	+++	+++	+++
	•Ship refueling ▸ When and in which port should each ship be refueled?	++	+++	++
	•Inventory ship routing ▸ What should the ship route and scheduling be so to maintain inventory levels within the desired interval?	-	+	+++
Operatio-nal	•Cruising speed selection ▸ What is the optimum speed that minimizes fuel consumption and also services all port calls within time windows?	++	++	++
	•Ship loading ▸ How should the cargo (e.g. containers) be placed inside the hold of the ship and also above the deck to maintain balance?	+++	+	+
	•Environmental routing ▸ How should the routes be constructed considering waves, tides, currents and bad weather forecasting?	++	+	+++

Problem relative importance: **- Practically none** **+ Low** **++ Medium** **+++ High**

Problems approached in this work

Appendix B – Procedure to create nodes of types 1, 2 and 3

Let the assignment constraints of ships $v \in V$ to contractual voyages $i \in CV$ and to spot voyages $r \in SV$ be represented by ACV_{vi} and ASV_{vr}, respectively:

$$ACV_{vi} = \begin{cases} 1 \text{ if ship } v \text{ may be assigned to contractual voyage } i; \\ 0 \text{ otherwise.} \end{cases}$$

$$ASV_{vr} = \begin{cases} 1 \text{ if ship } v \text{ may be assigned to spot voyage } r; \\ 0 \text{ otherwise.} \end{cases}$$

Figures B.1, B.2 and B.3 present the procedures to create the set of nodes of type 1, 2 and 3.

1. Let $N = \varnothing$ and $n = 0$;
2. For each contractual voyage $i \in CV$, do:
   2.1. Let $n = n + 1$ and $N = N \cup \{n\}$
   2.2. Let $type_n = 1$; $BC_n = \{i\}$; $BS_n = \varnothing$; $Ports_n = CVP_i$; $Demand_n = 0$;
   2.3. Calculate $CapMin_n$:
      2.3.1. Let $CapMin_n = 0$ and $tonnage = 0$;
      2.3.2. For each port call $p \in Ports_n$, do:
         2.3.2.1. Let $tonnage = tonnage - UCV_{ip} + LCV_{ip}$
         2.3.2.2. If $tonnage > CapMin_n$, then $CapMin_n = tonnage$
   2.4. For each ship $v \in V$, do:
      2.4.1. If $ACV_{vi} = 1$ and $NT_v \geq CapMin_n$, then

$$TotalTime_{vn} = \sum_{a=1}^{|CVP_i|} \left[LUT_{vo_i(a)} \left(LCV_{io_i(a)} + UCV_{io_i(a)} \right) \right] + \sum_{a=2}^{|CVP_i|} \left(Q_{vo_i(a)} + \frac{DP_{o_i(a-1)o_i(a)}}{SP_v} \right)$$

else $TotalTime_{vn} = \infty$

      2.4.2. If $BCV_i + TotalTime_{vn} <= TCV_i$ then

$$Profit_{vn} = RCV_i - RC_{vo_i(1)} \left[LUT_{vo_i(1)} \left(LCV_{io_i(1)} + UCV_{io_i(1)} \right) \right] - \sum_{a=2}^{|CVP_i|} RC_{vo_i(a)} \left[Q_{vo_i(a)} + LUT_{vo_i(a)} \left(LCV_{io_i(a)} + UCV_{io_i(a)} \right) \right] - VC_v \sum_{a=2}^{|CVP_i|} DP_{o_i(a-1)o_i(a)}$$

else $Profit_{vn} = -\infty$

      2.4.3. Let $BT_{vn} = BCV_i$; $ET_{vn} = \min(ECV_i, TCV_i - TotalTime_{vn})$

Figure B.1. Procedure to create voyage nodes of type 1

Steps 1 to 2.2 initialize sets N, BC_n, BS_n and $Ports_n$ and variables n, $type_n$ and $Demand_n$. $Demand_n = 0$ because all cargo loaded on a type 1 node is unloaded within the same node. Steps 2.3 to 2.3.2.2 calculate the maximum tonnage a ship will carry if it serves the node. This information also determines whether the ship has enough capacity to serve the node. If a ship v may serve node n (condition of step 2.4.1 is satisfied), $TotalTime_{vn}$ stores the sum of the queue, loading, unloading and traveling times of ship v serving node n. If ship v cannot serve node n, $TotalTime_{vn}$ assumes a very large number and $Profit_{vn}$ assumes a negative number on step 2.4.2. Step 2.4.2 calculates the profit when ship v serves node n. Finally, step 2.4.3 establishes the earliest and latest time ship v may begin service at node n to guarantee the time window feasibility.

The procedure of Figure B.2 has two blocks (1.1 and 1.2) that break down a single spot voyage into two type 2 nodes. The steps of block 1 constitute the loading port call of spot voyage r into a type 2 loading node ($Demand_n > 0$) and the steps of block 2 create another type 2 node from the unloading port call of the same spot voyage r ($Demand_n < 0$). Each block executes similar calculations of those of procedure of Figure B.1 to compute the values of $TotalTime_{vn}$ and $Profit_{vn}$.

1. For each spot voyage $r \in SV$, do:

1.1. Type 2 loading node:

1.1.1. Let $n = n + 1$; $N = N \cup \{n\}$; $type_n = 2$;

1.1.2. Let $BC_n = \varnothing$; $BS_n = \{r\}$; $Ports_n = \{LPSV_r\}$; $Demand_n = LSV_r$;

1.1.3. Let $CapMin_n = LSV_r$;

1.1.4. For each ship $v \in V$, do:

1.1.4.1. If $ASV_{vr} = 1$ and $NT_v \geq CapMin_n$, then $TotalTime_{vn} = LUT_{vLPSV_r} LSV_r$

else $TotalTime_{vn} = \infty$

1.1.4.2. If

$$BSV_r + TotalTime_{vn} + \left(\frac{DP_{LPSV_r UPSV_r}}{SP_v} + Q_{vUPSV_r} + LSV_r LUT_{vUPSV_r} \right) <= TSV_r,$$

then $Profit_{vn} = RSV_r - RC_{vLPSV_r}\left(LUT_{vLPSV_r} LSV_r\right)$

else $Profit_{vn} = -\infty$

1.1.4.3. Let $BT_{vn} = BSV_r$; Let $ET_{vn} = \min(ESV_r, TSV_r - TotalTime_{vn} - \left(\frac{DP_{LPSV_r UPSV_r}}{SP_v} + Q_{vUPSV_r} + LSV_r LUT_{vUPSV_r} \right))$

1.2. Type 2 unloading node:

1.2.1. Let $n = n + 1$; $N = N \cup \{n\}$; $type_n = 2$;

1.2.2. Let $BC_n = \varnothing$; $BS_n = \{r\}$; $Ports_n = \{UPSV_r\}$; $Demand_n = -LSV_r$;

1.2.3. Let $CapMin_n = 0$;

1.2.4. For each ship $v \in V$, do:

1.2.4.1. If $ASV_{vr} = 1$ then $TotalTime_{vn} = LUT_{vUPSV_r} LSV_r$, else $TotalTime_{vn} = \infty$

1.2.4.2. If $BSV_r + LSV_r LUT_{vLPSV_r} + \frac{DP_{LPSV_r UPSV_r}}{SP_v} + Q_{vUPSV_r} + TotalTime_{vn} <= TSV_r$,

then $Profit_{vn} = -RC_{vUPSV_r}\left(LUT_{vUPSV_r} LSV_r\right)$

else $Profit_{vn} = -\infty$

1.2.4.3. Let $BT_{vn} = BSV_r + LSV_r LUT_{vLPSV_r} + \frac{DP_{LPSV_r UPSV_r}}{SP_v} + Q_{vUPSV_r}$ and

let $ET_{vn} = TSV_r - TotalTime_{vn}$

Figure B.2. Procedure to create voyage nodes of type 2

1. Let $T2 = \{n \in N : type_n = 2\}$;
2. For each contractual voyage $i \in CV$, do:
   2.1. For each subset $ST2 \subseteq T2$ do:
      2.1.1. If it is feasible to create a sequence of port calls $CVP_i \cup \{\cup_{k \in ST2} Ports_k\}$ such that the first port call is $o_i(1)$ and the last port call is $o_i(|CVP_i|)$, then:
         2.1.1.1. Let $n = n + 1$ and $N = N \cup \{n\}$
         2.1.1.2. Let $type_n = 3$; $BC_n = \{i\}$; $BS_n = \{\cup_{k \in ST2} BS_k\}$; $Ports_n = CVP_i \cup \{\cup_{k \in ST2} Ports_k\}$;
         2.1.1.3. Let $Demand_n = \sum_{k \in ST2} Demand_k$;
         2.1.1.4. Let $CapMin_n = 0$ and $tonnage = 0$;
         2.1.1.5. For each port call $p \in Ports_n$, do:
            2.1.1.5.1. If $p \in CVP_i$ then let $tonnage = tonnage - UCV_{ip} + LCV_{ip}$
            else find k such that $p \in \{Ports_k : k \in ST2\}$ and let $tonnage = tonnage + Demand_k$
            2.1.1.5.2. If $tonnage > CapMin_n$, then $CapMin_n = tonnage$
         2.1.1.6. For each ship $v \in V$, do:
            2.1.1.6.1. If $ACV_{vi} = 1$ and $ASV_{vr} = 1$ for each $r \in BS_n$ and $NT_v \geq CapMin_n$, then let $TotalTime_{vn}$ be the sum of the queue (except for $Q_{voi(1)}$), loading/unloading and travelling times of the sequence of port calls of $Ports_n$;
            else let $TotalTime_{vn} = \infty$.
            2.1.1.6.2. Let $BT_{vn} = BCV_i$
            2.1.1.6.3. If $BT_{vn} + TotalTime_{vn} <= TCV_i$ then let $Profit_{vn}$ be the sum of the profits of the contractual and spot voyages (the revenue of a spot voyage is computed in the loading port call); else let $Profit_{vn} = -\infty$.
            2.1.1.6.4. Let $ET_{vn} = \min(ECV_i, TCV_i - TotalTime_{vn})$

Figure B.3. Procedure to create voyage nodes of type 3

The procedure of Figure B.3 merges one contractual voyage i with one or more spot voyages (subset $ST2$) into a type 3 node. The resulting type 3 node will always begin and end with the first and last port calls of contractual voyage i. Other spot voyages port calls may be linked before and/or after the type 3 node, as the examples shown in Figure 5.

Step 2.1.1 tests the feasibility of combining the contractual and spot voyages regarding time constraints $[BT_{vn}, ET_{vn}]$ and physical constraints (one should load before unloading). If feasible, the steps 2.1.1.1 to 2.1.1.3 create the type 3 node and steps 2.1.1.4 to 2.1.1.5.2 calculate the minimum capacity that a ship must have to be able to serve the node. Step 2.1.1.6.1 computes $TotalTime_{vn}$ and step 2.1.1.6.2 sets the minimum start time of service (BT_{vn}) of the new type 3 node. $Profit_{vn}$ is calculated in step 2.1.1.6.3 if it is feasible for ship v to serve node n. At last, step

2.1.1.6.4 evaluates the latest time window in which ship v may begin service at node n. The following points should be stressed about the procedure of Figure B.3:

- It is not necessary to test the feasibility of all subsets of step 2.1.1. For example, if there is not a feasible sequence of port calls with an element $k \in ST2$, all subsets that contain k will also be infeasible;
- There may be more than one feasible sequence of the same set of port calls on step 2.1.1. If more than one feasible sequence was found, a type 3 node should be created for each feasible sequence. For example, let cv_1, cv_2 and cv_3 be the sequence of port calls of a contractual voyage cv and L be the loading port call of a spot voyage s. Two type 3 nodes may be created, [cv_1-L-cv_2-cv_3] and [cv_1-cv_2-L-cv_3], each one with different $Profit_{vn}$, $CapMin_n$, $TotalTime_{vn}$ and ET_{vn} values;
- The time feasibility test $BT_{vn} + TotalTime_{vn} <= TCV_i$ of step 2.1.1.6.3 should be complemented with the time feasibility test of each spot voyage of the type 3 node, regarding both loading and unloading time constraints. Nodes that fail to satisfy the time feasibility test of each spot voyage are eliminated;
- Because time window constraints of both contractual and the spot voyages have to be satisfied in the type 3 node n, ET_{vn} of step 2.1.1.6.4 may be further restricted when also considering the time window constraints of each spot voyage merged into node n. For example, consider the route depicted in Figure 3 in which there is an unloading port call of a spot voyage in Brake. Suppose now that if a vessel starts service in Itajaí (the first port call of the contractual voyage) at the latest possible time of the contractual voyage time window, the vessel would finish service at Brake after the time window TSV_r of the spot voyage. In this case, ET_{vn} must be reduced to ensure that the vessel arrives at Brake with enough time to finish service before TSV_r.

Appendix C – Additional information on the mathematical models

For each $v \in V$ and m, $n \in N$, $m \neq n$, M_1 is calculated as:

$$BT_{vm} \geq TA_v + \frac{DP_{PA_v o_m(1)}}{SP_v} + Q_{vo_m(1)} - M_1(1 - xfirst_{vm}), \qquad \forall v \in V, \forall m \in N$$

$$BT_{vm} \geq TA_v + \frac{DP_{PA_v o_m(1)}}{SP_v} + Q_{vo_m(1)} - M_1, \qquad \forall v \in V, \forall m \in N$$

$$M_1 \geq -BT_{vm} + TA_v + \frac{DP_{PA_v o_m(1)}}{SP_v} + Q_{vo_m(1)}, \qquad \forall v \in V, \forall m \in N$$

$$M_1 = TA_v + \frac{DP_{PA_v o_m(1)}}{SP_v} + Q_{vo_m(1)} - BT_{vm}, \qquad \forall v \in V, \forall m \in N$$

M_2 is given by:

$$BT_{vn} \geq st_{vm} + TotalTime_{vm} + \frac{DP_{o_m(|Ports_m|)o_n(1)}}{SP_v} + Q_{vo_n(1)} - M_2(1 - x_{vmn}), \quad \forall v \in V, \forall m,n \in N, m \neq n$$

$$BT_{vn} \geq st_{vm} + TotalTime_{vm} + \frac{DP_{o_m(|Ports_m|)o_n(1)}}{SP_v} + Q_{vo_n(1)} - M_2 \quad \forall v \in V, \forall m,n \in N, m \neq n$$

$$M_2 \geq -BT_{vn} + st_{vm} + TotalTime_{vm} + \frac{DP_{o_m(|Ports_m|)o_n(1)}}{SP_v} + Q_{vo_n(1)} \quad \forall v \in V, \forall m,n \in N, m \neq n$$

Since the maximum value of st_{vm} is ET_{vm}, the inequality below guarantees a valid M_2:

$$M_2 \geq -BT_{vn} + ET_{vm} + TotalTime_{vm} + \frac{DP_{o_m(|Ports_m|)o_n(1)}}{SP_v} + Q_{vo_n(1)} \quad \forall v \in V, \forall m,n \in N, m \neq n$$

$$M_2 = -BT_{vn} + ET_{vm} + TotalTime_{vm} + \frac{DP_{o_m(|Ports_m|)o_n(1)}}{SP_v} + Q_{vo_n(1)} \forall v \in V, \forall m,n \in N, m \neq n$$

The real number M_3 is a large number that can be calculated for a ship $v \in V$ as follows:

$$NT_v \leq cap_{vm} - Demand_m + M_3(1 - x_{vmn}), \qquad \forall v \in V, \forall m,n \in N, m \neq n$$

$$NT_v \leq cap_{vm} - Demand_m + M_3, \qquad \forall v \in V, \forall m,n \in N, m \neq n$$

$$M_3 \geq NT_v - cap_{vm} + Demand_m, \qquad \forall v \in V, \forall m,n \in N, m \neq n$$

The minimum value that the variable cap_{vm} may assume is $CapMin_m$. Therefore, the following produces a valid inequality for M_3.

$$M_3 \geq NT_v - CapMin_m + Demand_m, \qquad \forall v \in V, \forall m,n \in N, m \neq n$$
$$M_3 = NT_v - CapMin_m + Demand_m, \qquad \forall v \in V, \forall m,n \in N, m \neq n$$

A valid M_4 is calculated below.

$$startTime_v \leq st_{vm} + M_4(1 - xfirst_{vm}), \qquad \forall v \in VCO, \forall m \in N$$
$$startTime_v - st_{vm} \leq M_4, \qquad \forall v \in VCO, \forall m \in N$$
$$M_4 \geq startTime_v - st_{vm}, \qquad \forall v \in VCO, \forall m \in N$$

The maximum value of $startTime_v$ is *latestTime* and the minimum value of st_{vm} is BT_{vm}. Thus, the following produces a valid M_4.

$$M_4 = latestTime - BT_{vm}, \qquad \forall v \in VCO, \forall m \in N$$

A valid M_5 is calculated below.

$$endTime_v \geq st_{vm} + TotalTime_{vm} - M_5(1 - xlast_{vm}), \qquad \forall v \in \{VCO \cup VHI\}, \forall m \in N$$
$$endTime_v - st_{vm} - TotalTime_{vm} \geq -M_5, \qquad \forall v \in \{VCO \cup VHI\}, \forall m \in N$$
$$M_5 \geq -endTime_v + st_{vm} + TotalTime_{vm}, \qquad \forall v \in \{VCO \cup VHI\}, \forall m \in N$$

The minimum value of $endTime_v$ is TA_v and the maximum value of st_{vm} is ET_{vm}. Thus, the following produces a valid M_5.

$$M_5 = -TA_v + ET_{vm} + TotalTime_{vm}, \qquad \forall v \in \{VCO \cup VHI\}, \forall m \in N$$

The following illustrates the requirement of constraints (22) for the model FULL.Cv with an example in which constraints (17), (18), (20) and (21) do not guarantee that $endTime_v \geq startTime_v$. Suppose the case in which $TA_v = BCO_v = 0$ and $ECO_v = \frac{3}{4} latestTime$. Replacing $TA_v = BCO_v = 0$ and $ECO_v = \frac{3}{4} latestTime$ in the expressions (17), (18), (20) and (21) results in:

$$startTime_v \geq 0, \qquad \forall v \in VCO$$

$$startTime_v \leq \frac{3}{4} latestTime, \qquad \forall v \in VCO$$

$$latestTime - endTime_v \geq 0, \qquad \forall v \in VCO$$

$$latestTime - endTime_v \leq \frac{3}{4} latestTime, \qquad \forall v \in VCO$$

Hence, $0 \leq startTime_v \leq \frac{3}{4} latestTime$, and, $\frac{1}{4} latestTime \leq endTime_v \leq latestTime$. Therefore, if $startTime_v = \frac{3}{4} latestTime$ and $endTime_v = \frac{1}{4} latestTime$, $endTime_v \leq startTime_v$.

Appendix D – Computational parameters and results

CPLEX was used with most of its parameters set to the default values (it was not the objective of this research to find the best CPLEX parameters to solve the mathematical models). However, for some test problems with a large number of variables and nodes, CPLEX was either unable to solve the root node because of lack of memory or unable to find the first feasible solution. In these cases, additional parameters were introduced, such as:

a) to limit cut passes to 4 (less cuts may reduce memory consumption);
b) to consider, at the beginning of the search, the variables $xfirst_{vm}$, $xlast_{vm}$ and x_{vmn} related *spot voyages* as zero (reduces the total number of variables while searching for the first feasible solution);
c) to increase the priority of selecting the variables $xfirst_{vm}$, $xlast_{vm}$ and x_{vmn} related to *contractual voyages* for branching and to set the preferable branching direction of these variables to 1 (also attempts to increase the probability of finding feasible solutions).

Since a feasible solution to a problem without spot voyages (e.g. FULL.Cv) is also feasible to a problem with spot voyages (e.g. FULL.CvSv), the parameters (b) and (c) encourage the method to branch down (toward zero) variables related to spot voyages and to branch up (toward one) variables related to contractual voyages.

Table D.1 presents the results of MIPE for the SIMPLE.Cv and SIMPLE.CvSv test problems and Table D.3 shows the results of MIPE for the FULL.Cv and FULL.CvSv test problems.

The first three columns of Table D.1 (Problem size columns) show, respectively, the total number of ships and the number of contractual and spot voyages of each test problem. The next three columns (Fleet deployment) present the summary of the fleet deployment plan of the best solution found: number of ships used, number of spot voyages served and the average number of port calls/ship. The following three columns (Solution Quality) contain the profit of the best feasible solution, the upper bound found by the branch-and-cut method and the solution gap as a percentage. A gap of 0.0% means that the optimal solution was found and a gap larger than 0.0%, for instance the gap of 4.3% for test problem SIMPLE.CvSv *f* means that there could be a solution better than the best feasible solution found (84420932), and this better solution would be at most 4.3% above the value of the best feasible solution: (88033541-84420932)/84420932=3612609/84420932=4.3%. The next four columns present the time (in seconds) to create the graph model, to create the MIP model, the time spent by the branch-and-cut method and finally the total time, which is the sum of the three previous columns. The incumbent columns present the time to find the best feasible solution (incumbent) and also the percentage of this time with respect to the total time. The final four columns show the total number of types 1, 2 and 3 nodes and also the total number of nodes.

Although there is a relationship between the number of nodes and number of ships and voyages, not every node is created because of the capacity and time window constraints. For example, one type 1 node may be created for each ship and contractual voyage. Therefore, for test problem SIMPLE.Cv *a*, which has 18 ships and 30 contractual voyages, there are, at most, 18 *x* 30 = 540 type 1 nodes. Table D.1 shows that, in this case, only 306 of the 540 type 1 nodes were created.

Table D.1. MIPE results – SIMPLE.Cv and SIMPLE.CvSv

		Problem size			Fleet deployment			Solution Quality			CPU Time (seconds)				Incumbent		Number of nodes			
		Ships	Cvs	Svs	# ships used	# Svs	Avg port calls/ ship	Profit	Upper Bound	Gap %	Node model	MIP model	Opt.	Total	Time to find (sec.)	% of total time	Type 1	Type 2	Type 3	Total
MIPE - Simple.Cv	a	18	30	0	8	0	43	3343742	3343742	**0.0%**	0.0	0.0	0.2	0.2	0.0	0%	306	0	0	306
	b	21	30	0	7	0	52	5410819	5410819	**0.0%**	0.0	0.0	0.7	0.7	0.6	87%	285	0	0	285
	c	28	35	0	7	0	66	40605550	40605550	**0.0%**	0.0	0.1	0.6	0.7	0.0	0%	524	0	0	524
	d	30	40	0	9	0	57	42236708	42236708	**0.0%**	0.0	0.1	1.6	1.7	1.6	96%	766	0	0	766
	e	34	45	0	10	0	52	18653004	18653004	**0.0%**	0.0	0.2	1.7	1.8	0.0	0%	1282	0	0	1282
	f	36	50	0	10	0	65	77698997	77705519	**0.0%**	0.0	0.2	2.7	2.9	2.7	94%	1383	0	0	1383
	g	36	60	0	12	0	65	44916802	44916802	**0.0%**	0.0	0.2	6.9	7.2	0.0	0%	1504	0	0	1504
	h	38	60	0	10	0	76	63535859	63535859	**0.0%**	0.0	0.3	9.8	10.2	9.8	97%	1667	0	0	1667
	i	38	60	0	12	0	61	62623784	62623784	**0.0%**	0.0	0.3	3.8	4.0	3.7	92%	1572	0	0	1572
	j	46	90	0	14	0	84	119112213	119123772	**0.0%**	0.1	1.3	61.7	63.1	61.5	98%	3256	0	0	3256
	k	46	90	0	16	0	72	113205382	113211803	**0.0%**	0.1	1.2	22.5	23.7	0.0	0%	2972	0	0	2972
	l	46	90	0	14	0	81	124150596	124150596	**0.0%**	0.1	1.3	61.3	62.7	0.0	0%	3253	0	0	3253
	m	50	110	0	18	0	78	137673738	137687309	**0.0%**	0.1	2.0	1046	1048	880	84%	3862	0	0	3862
	n	50	110	0	13	0	108	179342678	179358961	**0.0%**	0.1	1.9	257	259	247	96%	3663	0	0	3663
MIPE - Simple.CvSv	a	18	30	10	8	7	45	5826461	5826461	**0.0%**	0.2	0.2	5.2	5.6	4.9	88%	441	205	899	1545
	b	21	30	10	7	5	53	7796802	7796802	**0.0%**	0.1	0.1	7.1	7.3	5.2	72%	456	276	394	1126
	c	28	35	13	7	12	69	47185046	47185046	**0.0%**	2.1	0.5	38.2	40.8	0.0	0%	854	408	1207	2469
	d	30	40	13	9	11	59	47688363	47693054	**0.0%**	10.9	1.2	352	364	151	42%	1076	492	3008	4576
	e	34	45	16	10	15	55	25903562	25903562	**0.0%**	1.6	1.6	2505	2508	2478	99%	1758	835	3316	5909
	f	36	50	16	10	12	67	84420932	88033541	4.3%	7.1	2.8	86439	86449	86449	100%	1933	936	6318	9187
	g	36	60	19	12	15	68	50760146	56193535	10.7%	433	6.5	86330	86770	52764	61%	2076	947	16089	19112
	h	38	60	19	10	10	78	68175652	70130569	2.9%	3.9	3.7	86740	86748	31639	36%	2134	684	6967	9785
	i	38	60	19	12	16	64	68806103	68807958	**0.0%**	86.3	4.0	1331	1421	1294	91%	2243	987	5816	9046
	j	46	90	28	om	om	om	120747447	130231788	7.9%	48.1	23.9	om	om	31805	om	4262	1619	21320	27201
	k	46	90	28	16	20	74	118862181	120006501	1.0%	107	14.5	86259	86381	71181	82%	4061	1672	17977	23710
	l	46	90	28	om	om	om	om	om	om	95.8	23.9	om	om	om	om	4309	1779	32797	38885
	m	50	110	34	om	om	om	om	om	om	831	42.9	om	om	om	om	5147	2028	41181	48356
	n	50	110	34	om	om	om	om	om	om	34.9	27.7	om	om	om	om	5102	2589	25768	33459

om: execution stopped because the program ran out of memory.

For the SIMPLE.Cv test problems, MIPE was able to find the optimal solutions for all problems in at most 1048 seconds (roughly 17 minutes). However, for the SIMPLE.CvSv test problems, MIPE only found the optimal solution for six problems (*a*, *b*, *c*, *d*, *e* and *i*). For the rest of the test problems, there is no guarantee that the feasible solutions that were found are optimal since gaps range between 1% and 11%. Test problems *j*, *l*, *m* and *n* were not completed solved because there was not enough memory to store the minimum branch and bound tree. Nevertheless, CPLEX heuristics found a feasible solution for test problem *j* before the process was killed due to the excess of memory consumption. The analysis of these four test problems suggests that memory problems start to appear when the total number of nodes is around 30,000.

The results of Table D.1 also show that the profit of the SIMPLE.CvSv is greater than that of SIMPLE.Cv for all test problems from *a* to *k*. On average, the profit of the test problems with spot voyages are 21% greater than the profit of the test problems without spot voyages. The main cause of the greater profit of the SIMPLE.CvSv test problems is that the additional spot voyages revenues are captured with a small increase of the overall costs. Table D.2 illustrates this situation for the test problem *e*.

Table D.2. Revenues and costs of test problem *e* - SIMPLE.Cv and SIMPLE.CvSv

	Simple.Cv (A)	**Simple.CvSv (B)**	**Δ (%): [(B)-(A)]/(A)**
Revenue	**108827782**	**117621304**	**8%**
Cv	108827782	108827782	0%
Sv	0	8793522	-
(-) Costs	**90174778**	**91717742**	**2%**
(=) Profit	**18653004**	**25903562**	**39%**

The first two columns of Table D.2 show the financial information of the optimal solutions of test problems *e* SIMPLE.Cv and SIMPLE.CvSv. The last column presents the percentage difference between the previous two columns. The interesting conclusion extracted from the data of the last column is that the eight percentage increase in revenue, caused by the additional spot voyage revenue of SIMPLE.CvSv, produces a profit increase of 39%.

For the FULL.Cv MIPE found the optimal solutions for the first 11 test problems and for the FULL.CvSv the optimal solution was found for only the first three problems (Table D.3). When the optimal solution was not found (or proven), the execution of MIPE stopped at the time limit of 24 hours (86400 seconds). The largest solution gaps were 4.2% for the FULL.Cv set of test problems and 48.9% for the FULL.CvSv test problems.

Table D.3. MIPE results – FULL.Cv and FULL.CvSv

		Problem size			Fleet			Solution Quality			CPU Time (seconds)				Incumbent		Number of nodes			
		Ships	Cvs	Svs	# ships used	# Svs	Avg port calls/ ship	Profit	Upper Bound	Gap %	Node model	MIP model	Opt.	Total	Time to find (sec.)	% of total time	Type 1	Type 2	Type 3	Total
MIPE - Full.Cv	a	24	30	0	9	0	39	51592988	51596819	**0.0%**	0.0	0.0	41.3	41.3	40.2	97%	377	0	0	377
	b	25	30	0	12	0	30	107105606	107114390	**0.0%**	0.0	0.0	25.8	25.8	25.0	97%	400	0	0	400
	c	34	35	0	10	0	46	109397725	109397725	**0.0%**	0.0	0.1	33.6	33.7	3.5	10%	615	0	0	615
	d	42	40	0	16	0	32	125692600	125692600	**0.0%**	0.0	0.1	55.3	55.4	53.5	97%	970	0	0	970
	e	41	45	0	13	0	40	103860019	103860019	**0.0%**	0.0	0.2	64.9	65.1	64.9	100%	1447	0	0	1447
	f	56	50	0	15	0	43	165357931	165374463	**0.0%**	0.0	0.3	1576	1577	1486	94%	1859	0	0	1859
	g	48	60	0	21	0	37	160483860	160498539	**0.0%**	0.0	0.3	368	368	366	99%	1871	0	0	1871
	h	49	60	0	17	0	45	141380920	141395010	**0.0%**	0.0	0.4	1661	1661	1591	96%	2066	0	0	2066
	i	60	60	0	19	0	39	202918389	202938679	**0.0%**	0.0	0.5	30222	30223	20714	69%	2308	0	0	2308
	j	70	90	0	23	0	51	256347098	256372708	**0.0%**	0.1	2.0	42407	42409	41121	97%	5015	0	0	5015
	k	70	90	0	23	0	50	239460694	239484284	**0.0%**	0.1	1.8	17903	17905	17905	100%	4608	0	0	4608
	l	70	90	0	24	0	47	248715239	257872382	3.7%	0.1	2.1	86381	86383	86383	100%	5009	0	0	5009
	m	82	110	0	24	0	59	319890971	333177392	4.2%	0.1	3.0	86395	86398	82045	95%	6114	0	0	6114
	n	82	110	0	26	0	54	304723251	309550597	1.6%	0.1	2.8	86397	86400	85999	100%	5616	0	0	5616
MIPE - Full.CvSv	a	24	30	10	9	6	40	53797761	53803140	**0.0%**	0.2	0.3	13548	13549	11895	88%	555	274	1178	2007
	b	25	30	10	10	5	37	110047995	110058947	**0.0%**	0.1	0.1	8867	8867	8566	97%	602	316	493	1411
	c	34	35	13	10	9	48	114180627	114191978	**0.0%**	2.7	0.7	16672	16675	16155	97%	1017	498	1469	2984
	d	42	40	13	15	10	35	129245081	143149306	10.8%	15.5	1.5	86504	86521	84262	97%	1400	681	3834	5915
	e*	41	45	16	13	2	40	102681169	127234425	23.9%	1.8	1.8	86725	86729	40938	47%	2013	984	3730	6727
	f*	56	50	16	14	12	48	133105070	198208257	48.9%	9.3	3.7	86417	86430	60454	70%	2699	1355	8406	12460
	g*	48	60	19	20	3	40	145243552	197463349	36.0%	542	8.0	86446	86996	85585	98%	2635	1247	20292	24174
	h*	49	60	19	15	2	51	122450002	175585155	43.4%	5.3	4.8	86430	86441	4017	5%	2677	880	9025	12582
	i*	60	60	19	18	6	42	189643988	236600657	24.8%	107	5.7	86416	86529	5227	6%	3360	1556	8638	13554
	j	70	90	28	om	om	om	om	om	om	69.8	35.9	om	om	om	om	6542	2488	31742	40772
	k*	70	90	28	om	om	om	216051861	284444149	31.7%	150	22.4	om	om	50740	om	6266	2553	28090	36909
	l	70	90	28	om	om	om	om	om	om	149	36.0	om	om	om	om	6604	2713	48995	58312
	m	82	110	34	om	om	om	om	om	om	1235	61.9	om	om	om	om	8219	3306	60054	71579
	n	82	110	34	om	om	om	om	om	om	52.1	41.1	om	om	om	om	7975	4216	38926	51117

om: execution stopped because the program ran out of memory.

**: feasible solution was only found when the limit of cut passes was set to 4 and when warm startup and priorities values were set for the variables $xfirst_{vm}$, $xlast_{vm}$ and x_{vmn}.*

Memory problems were severe for the FULL.CvSv. It was not possible to create a complete branch and bound tree for 5 of the 14 test problems because of the lack of CPU memory. Also, it was only possible to find feasible solutions for problems *e* to *i* and *k* if the limit of cut passes was set to 4 (to save memory) and if the warm startup and priority values were set to variables $xfirst_{vm}$, $xlast_{vm}$ and x_{vmn}.

For FULL.Cv, test problems *j*, *l*, *m* and *n* were the ones with the highest time to find the incumbent. Figure D.1 analyses the impact on the solution quality if the time limit of 24 hours is reduced.

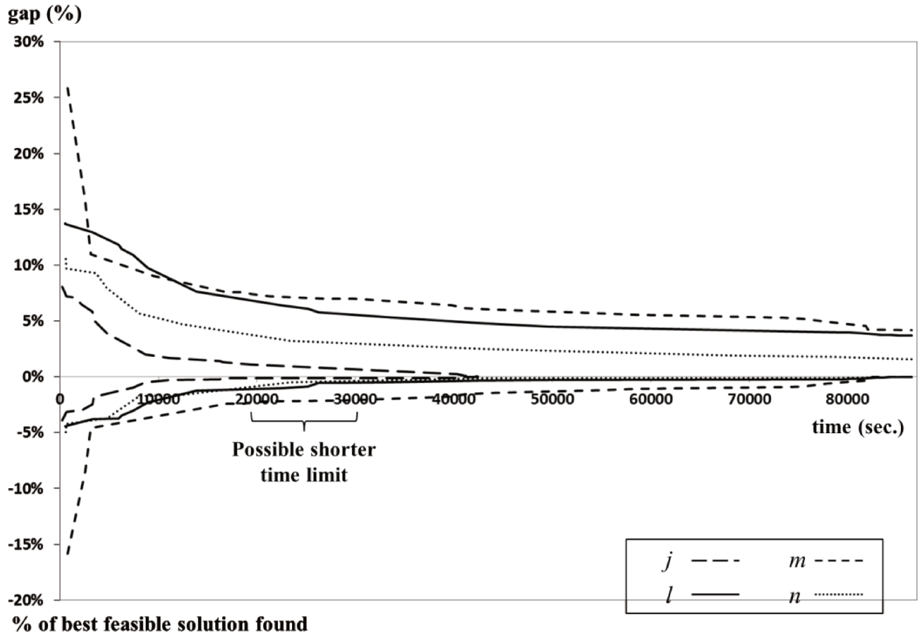


Figure D.1. Solution quality of 4 test problems of MIPE FULL.Cv

The upper part of the graph (positive y-axis) represents the gap between the best node and the best feasible solution found so far. The lower part of the graph (negative y-axis) represents the % of the best feasible solution found so far when compared to the best feasible solution found at the end of the total running time. In other words, the lower part measures how far is the quality of the current feasible solution to the best feasible solution to be found during the 24 hours. Since optimality was only proven for test problem *j*, only the *j* upper line touches the x-axis at about 42,000 seconds. Considering this sample of 4 problems, one may conclude that a time limit between 20,000 and 30,000 seconds could be enough to find high quality solutions with MIPE for the FULL.Cv test problems.

The profits of the FULL.CvSv test problems *a* to *d* are, on average, 3.5% greater than the profits of the FULL.Cv test problems. However, the profit of the large test problems (test problems *e* to *i* and *k* with % gap greater than 10.8%) is on average, 9.9% lower than that of the respective FULL.Cv test problems. Therefore, MIPE was unable to find high quality solutions for the larger FULL.CvSv test problems.

Table D.4 shows the importance of the application of the dominance criteria (see section 4.6) when creating the type 3 nodes. The first 5 columns show, respectively, the total number of nodes, profit, upper bound, percentage gap and total CPU time for MIPE with dominance criteria. The next five columns of Table D.4 show the same information for MIPE without dominance criteria. The last 3 columns show the percentage difference between MIPE without and with dominance criteria regarding total number of nodes, profit and total CPU time.

Table D.4. Comparison of the results of MIPE with and without dominance criteria

		A. With Dominance Criteria					B. Without Dominance Criteria					Δ%: (B-A)*100/A		
		Total # of nodes	Profit	Upper Bound	Gap %	Total CPU Time (sec.)	Total # of nodes	Profit	Upper Bound	Gap %	Total CPU Time (sec.)	Total # of nodes	Profit	Total CPU Time (sec.)
MIPE - Simple.CvSv	a	1545	5826461	5826461	0.0%	5.6	3236	5826461	5826461	0.0%	35.1	109%	0.0%	526%
	b	1126	7796802	7796802	0.0%	7.3	2142	7796802	7796802	0.0%	10.6	90%	0.0%	45.7%
	c	2469	47185046	47185046	0.0%	40.8	15198	47185046	47187296	0.0%	5882	516%	0.0%	14305%
	d	4576	47688363	47693054	0.0%	364	240447	om	om	om	om	5155%	na	na
	e	5909	25903562	25903562	0.0%	2508	15987	25903562	25903562	0.0%	4109	171%	0.0%	63.8%
	f	9187	84420932	88033541	4.3%	86449	45654	om	om	om	om	397%	na	na
	g	19112	50760146	56193535	10.7%	86770	1317681	om	om	om	om	6795%	na	na
	h	9785	68175652	70130569	2.9%	86748	71283	om	om	om	om	628%	na	na
	i	9046	68806103	68807958	0.0%	1421	123302	om	om	om	om	1263%	na	na
	j	27201	120747447	130231788	7.9%	om	162348	om	om	om	om	497%	na	na
	k	23710	118862181	120006501	1.0%	86381	516750	om	om	om	om	2079%	na	na
	l	38885	om	om	om	om	352097	om	om	om	om	805%	na	na
	m	48356	om	om	om	om	4897736	om	om	om	om	10028%	na	na
	n	33459	om	om	om	om	152315	om	om	om	om	355%	na	na
MIPE - Full.CvSv	a	2007	53797761	53803140	0.0%	13549	4217	53259580	57780618	8.5%	86472	110%	-1.0%	538%
	b	1411	110047995	110058947	0.0%	8867	2633	109824156	120032230	9.3%	9060	87%	-0.2%	2.2%
	c	2984	114180627	114191978	0.0%	16675	19189	109356934	122818349	12.3%	86857	543%	-4.2%	421%
	d	5915	129245081	143149306	10.8%	86521	312501	om	om	om	om	5183%	na	na
	e	6727	102681169	127234425	23.9%	86729	17926	91779546	129052988	40.6%	86580	166%	-10.6%	-0.2%
	f	12460	133105070	198208257	48.9%	86430	62979	om	om	om	om	405%	na	na
	g	24174	145243552	197463349	36.0%	86996	1678706	om	om	om	om	6844%	na	na
	h	12582	122450002	175585155	43.4%	86441	96177	om	om	om	om	664%	na	na
	i	13554	189643988	236600657	24.8%	86529	164474	om	om	om	om	1113%	na	na
	j	40772	om	om	om	om	237381	om	om	om	om	482%	na	na
	k	36909	216051861	284444149	31.7%	om	734118	om	om	om	om	1889%	na	na
	l	58312	om	om	om	om	528436	om	om	om	om	806%	na	na
	m	71579	om	om	om	om	7059335	om	om	om	om	9762%	na	na
	n	51117	om	om	om	om	222082	om	om	om	om	334%	na	na

om: execution stopped because the program ran out of memory
na: not available because the program ran out of memory

Table D.4 presents that 11 test problems that are solved with the dominance criteria are not solved without dominance criteria. The explanation for this result is the greater number of nodes of the models without dominance criteria. For example, the model of test problem SIMPLE.CvSv *d* without dominance criteria has about 5.000% more nodes than that with dominance criteria. The following conclusions are drawn for the other 8 test problems that are solved without dominance criteria:

- For the four SIMPLE.CvSv test problems that were solved to optimality, the increase of the total CPU time ranges from 46% to 14305% ;

- For the four FULL.CvSv test problems, the profit of the best feasible solution of the model without dominance criteria is always worse than that with dominance criteria. The worst profit difference is -10.6% for test problem *e*;
- The computational time without dominance criteria for the FULL.CvSv test problems *a* and *c* is at least 400% greater than that with dominance criteria. The time increase for test problem *b* is only 2.2% because CPLEX aborted the optimization of the model without dominance criteria when the size of the branch-and-bound exceeded the limit of 12Gb.

To overcome the memory problems and also to investigate if good quality solutions could be found in a shorter time, especially for the FULL.CvSv, the MIPBN00 and MIPBN15 model variants, which make use of a graph with a smaller number of nodes, were tested on SIMPLE.CvSv and FULL.CvSv test problems. Table D.5 compares the results of these methods to those of MIPE.

The Gap % column of Table D.5 shows the gap between the value of the best feasible solution found by MIPBN00 and MIPBN15 and the best solution found by MIPE (Tables D.1 and D.3). The last three columns of Table D.5 present the profit difference, in percentage, between each method of the column (MIPE, MIPBN00 and MIPBN15) and the profit of the best feasible solution (second column of the group of 4 columns of Best Feasible Solution).

For SIMPLE.CvSv, MIPE and MIPBN15 produce the best results (except for test problem *j* in which the best result was found by MIPBN00). As a general rule, MIPE is the preferred choice for smaller problems (*a* to *i*) and MIPBN15 for larger problems (*k* to *n*). For FULL.CvSv, MIPE produces the best results only to very small problems, such as *a* to *c*. For the larger FULL.CvSv test problems, MIPBN00 usually generates better results than MIPBN15.

The results of Tables D.3 and D.5 also show that the best profit obtained in Table D.5 for the FULL.CvSv test problems *a* to *i* is, on average, 2.3% greater than that of the respective test problems of the FULL.Cv of Table D.3. However, the results of the FULL.Cv test problems *j* to *n* of Table D.3 (without spot voyages) are still better than those of Table D.5 (with spot voyages). This suggests that large FULL.CvSv problems (e.g. fleet between 40 and 80 ships) are too complex for the proposed MIP models.

Table D.5. Comparison of results between MIPE, MIPBN00 and MIPBN15

		Problem size			Best Feasible Solution				Δ % of best feasible solution		
		Ships	Cvs	Svs	Methods	Profit	Gap %	Time to find	MIPE	MIPBN00	MIPBN15
Simple.CvSv	a	18	30	10	MIPE,MIPBN15	5826461	0.0%	4.9	**0.0%**	-16.6%	**0.0%**
	b	21	30	10	MIPE,MIPBN15	7796802	0.0%	5.2	**0.0%**	-15.6%	**0.0%**
	c	28	35	13	MIPE	47185046	0.0%	0.0	**0.0%**	-1.8%	0.0%
	d	30	40	13	MIPE	47688363	0.0%	151	**0.0%**	-0.5%	0.0%
	e	34	45	16	MIPE	25903562	0.0%	2478	**0.0%**	-4.0%	-0.2%
	f	36	50	16	MIPBN15	84430337	4.3%	64292	0.0%	-1.6%	**0.0%**
	g	36	60	19	MIPBN15	51399688	9.3%	84891	-1.2%	-1.6%	**0.0%**
	h	38	60	19	MIPE	68175652	2.9%	31639	**0.0%**	-2.5%	-0.2%
	i	38	60	19	MIPE	68806103	0.0%	1294	**0.0%**	-2.2%	-0.1%
	j	46	90	28	MIPBN00	126615871	2.9%	62265	-4.6%	**0.0%**	-2.0%
	k	46	90	28	MIPBN15	119226796	0.7%	78989	-0.3%	-0.9%	**0.0%**
	l	46	90	28	MIPBN15	132634885	na	75466	na	-4.6%	**0.0%**
	m	50	110	34	MIPBN15	147522566	na	58385	na	-0.1%	**0.0%**
	n	50	110	34	MIPBN15	184565356	na	20039	na	-0.9%	**0.0%**
Full.CvSv	a	24	30	10	MIPE,MIPBN15	53797761	0.0%	11895	**0.0%**	-1.0%	**0.0%**
	b	25	30	10	MIPE,MIPBN15	110047995	0.0%	8566	**0.0%**	-1.8%	**0.0%**
	c	34	35	13	MIPE	114180627	0.0%	16155	**0.0%**	-2.1%	-0.1%
	d	42	40	13	MIPBN15	130331112	9.8%	86123	-0.8%	-0.8%	**0.0%**
	e	41	45	16	MIPBN00	105426295	20.7%	77557	-2.6%	**0.0%**	-1.5%
	f	56	50	16	MIPBN00	166037720	19.4%	82503	-19.8%	**0.0%**	-7.7%
	g	48	60	19	MIPBN00	163554256	20.7%	84094	-11.2%	**0.0%**	-7.6%
	h	49	60	19	MIPBN00	142974298	22.8%	80390	-14.4%	**0.0%**	-5.5%
	i	60	60	19	MIPBN00	204197546	15.9%	84267	-7.1%	**0.0%**	-5.3%
	j	70	90	28	MIPBN00	203134480	na	34037	na	**0.0%**	na
	k	70	90	28	MIPBN00	223514635	27.3%	53044	-3.3%	**0.0%**	-3.0%
	l	70	90	28	MIPBN00	199436913	na	86383	na	**0.0%**	na
	m	82	110	34	MIPBN00	277146850	na	86414	na	**0.0%**	na
	n	82	110	34	na	na	na	na	na	na	na

na: not available because the program ran out of memory or because no feasible solution was found.

Table D.6 shows the fleet composition of the best results of the FULL problems. The sum of each row of the six columns of Table D.6 that present the number of ships per type is equal to the number of ships of the first column. The comparison between FULL.Cv and SIMPLE.Cv and between FULL.CvSv and SIMPLE.CvSv shows that FULL results are always (at least 50%) better than the SIMPLE results. These results are explained by the use of cheaper hired ships (instead of own ships) and the capture of additional charter revenues, as show in the last columns of Table D.6.

Table D.6. Fleet composition of the best results of the FULL problems

		Problem size			Best Feasible Solution		Number of own ships				Number of hired ships		
							Used to serve voyages		Not used to serve voyages				
		Ships	Cvs	Svs	Methods	Profit	Not chartered	Chartered	Chartered	Laid-up	Used	Not used	Chartered revenue
Full.Cv	a	24	30	0	MIPE	51592988	5	4	2	7	0	6	76947721
	b	25	30	0	MIPE	107105606	3	8	2	8	1	3	150331360
	c	34	35	0	MIPE	109397725	3	7	0	18	0	6	110866161
	d	42	40	0	MIPE	125692600	3	12	1	14	1	11	148617663
	e	41	45	0	MIPE	103860019	4	8	0	22	1	6	159854582
	f	56	50	0	MIPE	165357931	7	8	3	18	0	20	189569764
	g	48	60	0	MIPE	160483860	5	12	1	18	4	8	201890128
	h	49	60	0	MIPE	141380920	6	9	0	23	2	9	164556026
	i	60	60	0	MIPE	202918389	4	12	1	21	3	19	223794786
	j	70	90	0	MIPE	256347098	6	14	1	25	3	21	246767070
	k	70	90	0	MIPE	239460694	5	15	0	26	3	21	211544899
	l	70	90	0	MIPE	248715239	5	15	1	25	4	20	241468064
	m	82	110	0	MIPE	319890971	4	17	0	29	3	29	279192082
	n	82	110	0	MIPE	304723251	5	17	0	28	3	29	237124483
Full.CvSv	a	24	30	10	MIPE,MIPBN15	53797761	5	4	2	7	0	6	76947721
	b	25	30	10	MIPE,MIPBN15	110047995	3	6	4	8	1	3	146317712
	c	34	35	13	MIPE	114180627	3	7	0	18	0	6	110871249
	d	42	40	13	MIPBN15	130331112	3	12	1	14	1	11	150218828
	e	41	45	16	MIPBN00	105426295	4	8	0	22	2	5	159854582
	f	56	50	16	MIPBN00	166037720	7	7	4	18	1	19	189607342
	g	48	60	19	MIPBN00	163554256	5	12	1	18	5	7	205071906
	h	49	60	19	MIPBN00	142974298	7	9	0	22	1	10	165133666
	i	60	60	19	MIPBN00	204197546	3	11	2	22	4	18	223304700
	j	70	90	28	MIPBN00	203134480	2	15	0	29	4	20	184164240
	k	70	90	28	MIPBN00	223514635	3	15	0	28	2	22	182385710
	l	70	90	28	MIPBN00	199436913	2	15	1	28	4	20	180883411
	m	82	110	34	MIPBN00	277146850	2	17	0	31	4	28	225707274
	n	82	110	34	na	na	na	na	na	na	na	na	na

na: not available because the program ran out of memory or because no feasible solution was found.

3. PAPER B - TABU SEARCH FOR FLEET DEPLOYMENT AND ROUTING IN LINER SHIPPING

Tabu search for fleet deployment in liner shipping

Abstract

We address a tactical planning problem faced by many liner shipping companies that have committed contractual voyages while trying to serve optional spot voyages to increase its revenue over the medium-term horizon. The decisions include the number and type of vessels deployed, the assignment of vessels to contractual and spot voyages and the determination of vessel routes and schedules in order to maximize profit. A tabu search algorithm with a candidate list, a tree search and a pool of elite and diverse solutions is proposed in order to solve a set of benchmark instances of the problem. The results obtained by tabu search are compared to optimal and suboptimal solutions yielded by the CPLEX solver to mixed integer programming formulations of the problem.

***Keywords**: Logistics, maritime transportation, liner shipping, routing, heuristics, tabu search.*

1. Introduction

A common classification of shipping companies in modes of maritime transportation is liner, tramp and industrial operations (Lawrence [19]). A liner shipping company operates similar to a bus line, following a determined and published route. The liner company pickups and delivers client cargoes, e.g. containers, along the route analogous to the hop-on and hop-off of passengers in a bus line. A tramp shipping company does not have a predefined route to follow, the route is constructed and executed as new transport demands, such as dry bulk, gas or chemicals arrive. The analogy here is that a tramp company operates like a taxicab, picking up and delivering passengers while it is on the way. The tramp shipping operation may be full shipload, as of a taxicab with a single passenger, or parcel loads, as of a shared taxicab. The industrial operation is common on a verticalized company, such as mining companies that own or control both the cargo and the ships.

This work deals with the Fleet Size and Mix Routing Tactical Problem (FSMRTP) in liner shipping that could also be extended to tramp shipping operations. For the medium term planning horizon, for example, from six months to a year, fleet size decisions determine the necessary number of ships and select the types of ships to use, including the evaluation of the following alternatives: to lay-up a ship, i.e., to moor it in a protected anchorage or berth with most onboard systems shut down to reduce costs, to hire or to charter that means hiring a ship from another company or renting a ship to another company, respectively, for a determined period of time and price. Therefore, the objective of the FSMRTP is to maximize profit through the determination of (i) the number and type of its own ships, as well as the number and type of laid-up and hired and charter ships, (ii) the set of spot voyages to be served during the planning horizon and (iii) the ship routes and schedules. The FSMRTP is modeled in four problem variants shown in Table 1.

Table 1. Problem models

Model	Voyages		Ships		
	Contractual	Spot	Laid-up	Chartered	Hired
SIMPLE.Cv	✓				
SIMPLE.CvSv	✓	✓			
FULL.Cv	✓		✓	✓	✓
FULL.CvSv	✓	✓	✓	✓	✓

The SIMPLE problem models only consider owned or controlled ships, and these ships may not be either laid-up or chartered. There are two models of SIMPLE, one that serves only contractual voyages (SIMPLE.Cv) and one that serves contractual voyages and may serve spot voyages (SIMPLE.CvSv). The FULL problem models consider owner's ships that may be laid-up and/or chartered, and hired ships. There are also two models of FULL, one that serves only contractual voyages (FULL.Cv) and one that serves contractual voyages and may serve spot voyages (FULL.CvSv).

The remainder of the paper is organized as follows. Section 2 presents a literature review and Section 3 describes the problem. The tabu search algorithm is given in Section 4. The description of the test problems and the results of the computational experiments are shown in Section 5. Finally, conclusions and suggestions for future work are discussed in Section 6.

2. Literature review

This section presents the related work to the FSMRTP and also the research that has applied heuristics to other maritime problems such as tramp shipping. For a thorough review on the optimization of maritime problems, we refer to Ronen [22,23], Christiansen et al. [7,8] and, more recently, Christiansen et al. [10].

Brønmo et al. [4] present a multi-start local search heuristic for the tramp shipping scheduling problem. The heuristic searches for a maximum profit solution that solves the pickup and delivery problem of bulk cargo in the tramp market. The voyage of a ship may consist of more than one pickup or delivery points and may be executed by own or hired ships. Spot voyages are accepted if feasible and profitable. The authors consider a heterogeneous fleet with different cost structures and load capacities. In addition, the fleet size and mix is given and remains unchanged during the planning horizon. The search may use up to 2 intra-route neighborhoods (1-resequence and 2-resequence) and 3 inter-route neighborhoods (reassign, 2-interchange and 3-interchange). The heuristic solutions are compared to the set partition benchmarks and the result was a less than 2.2% average optimality gap.

More recently, Norstad et al. [20] introduce a mathematical formulation of the tramp ship routing and scheduling problem with speed optimization and use a multi-start local search heuristic, based on Brønmo et al. [4], to solve it. After each new route is constructed, the speed optimization subproblem is solved by one of the two exact algorithms presented: discretizing arrival times that may be applied to any fuel consumption function, including functions that depend on shipload, and a recursive smoothing algorithm which is only applied to convex fuel consumption functions. The authors conclude that speed optimization increases profit because it makes possible for the shipping company to accept more spot cargoes, increasing the speed as needed, and reducing fuel consumption per distance by lowering the speed when possible.

Fagerholt et al. [12] apply a multi-start local search heuristic adapted from Brønmo et al. [4] to a case study of the fleet deployment problem for a liner shipping company that provides ro-ro vehicle transportation services. The objective is to assign a set of owned and hired ships to a set of voyages considering the time window of each voyage. The cargo of each voyage is not explicitly considered, such as how many vehicles to load and unload in each port call, because this information is unknown for the whole planning horizon. Therefore, for each pair "ship *x* voyage" an estimate of cost and service time is associated. The multi-start local search heuristic was embedded in a decision support system and produced an improvement between 2% and 10% of the manual planning solutions of the transportation company.

Genetic algorithms and large neighborhood search have been applied to maritime problems. Karlaftis et al. [15] use a genetic algorithm for the route scheduling problem of a homogeneous fleet of containerships performing short-distance pickups and deliveries between hub-and-spoke ports. The problem is inspired on the supply network between Greece mainland and islands, including the pickup cargo from islands and time deadlines in ship arrivals. The genetic algorithm is tested for routing a small fleet with 6-7 ships that operates in the ports of Piraeus and the ports of other 25 islands, and produces optimal routes in a low (2 to 3 minutes) execution time. Christiansen et al. [9] develop a constructive heuristic and a genetic algorithm to approach the maritime inventory routing problem of a major Norwegian cement producer. The company has two factories which produce up to 10 cement grades products, and 28 consumption ports. Each consumption port may have up to 5 different silos, and each product must be stored separately of each other in different silos. The company controls a heterogeneous fleet of 5 ships through long term contracts and determines the ships routes and schedules in 2 weeks planning horizon. The solution method must determine the ships routes and the loading and unloading quantities at each port, considering the inventory lower and upper limits of each silo, customer demands and transportation costs. The results were compared with those obtained manually by the company, and they proved to be of better quality.

Korsvik et al. [18] propose a large neighborhood search heuristic for the tramp ship routing and scheduling problem with split loads. This problem admits that the demand, corresponding to the total amount of cargo to be transported from the port of origin to the port of destination, may be served by more than one ship and voyage. The objective is to maximize profit and to serve all mandatory demands. Spot cargo may be served if feasible and profitable. The large neighborhood search consists of a descent local search with four local operators – reassign, interchange, 1-split or merge and 2-split or merge – followed by a destroy and repair algorithm. The last two operators of the DLS are particularly designed to deal with split loads. The authors conclude that the introduction of split loads increases the fleet capacity utilization and, consequently, produces more profit.

Korsvik et al. [17] propose a tabu search heuristic for the ship routing and scheduling problem with a similar structure of the tabu search developed by Cordeau et al. [11] for the vehicle routing problem with time windows. The method considers infeasible solutions during the search, which violate capacity and/or time window constraints, and also penalizes the objective function when a cargo is not served (mandatory and/or optional cargoes). A periodic intra-route move is performed to improve the current solution. In a final intensification phase, a descent local search with move operators suggested in Brønmo et al. [4] is employed. The results show that the tabu search produces much better solutions than those of Brønmo et al. [4], particularly for the large and tightly constrained test cases.

Another tabu search heuristic is presented in Korsvik and Fagerholt [16] to deal with the ship routing and scheduling problem with flexible cargo quantities. This problem is also found in Brønmo et al. [5] and is common among tramp shipping companies transporting bulk products.

The tabu search penalizes the objective function when a mandatory and/or optional cargo is not served, and when a cargo is served outside the feasible quantity interval. A solution of the tabu search is defined as a set of pairs "ship x cargo" assignments and the neighborhood of the solution is defined as any solution that can be found through the following two moves: reassignment of a cargo to another ship and interchange of one cargo of each ship with one another. Once a cargo i is removed from a ship v, the cargo i is forbidden to be assigned again to ship v for some of the following iterations of the search. To employ a diversification strategy, the objective function is penalized when a pair "ship x cargo" has been inserted into the solutions too frequently. Another diversification strategy is used after each w iterations: the best non-tabu reassignment move is applied to the pair "ship x cargo" that has been present in the solution for the most consecutive number of iterations. Intra-route moves are also performed as an intensification tool at every k iterations or whenever a new incumbent solution has been found. Computational tests show that the tabu search produces high quality solutions in significantly less time compared with the column generation approaches of Brønmo et al. [6].

Álvarez [1] presents a mixed integer programming model and an algorithm to tackle the fleet deployment and routing problem in liner shipping. The objective is to minimize the operating expenses of the liner company over a tactical planning horizon. The solution approach is separated into two tiers. The higher tier is governed by a tabu search algorithm that determines the number of ships assigned to each run, which is a combination of ship type, speed and service. The lower tier is a pure multi-commodity flow problem which is solved by the commercial solver CPLEX. Computational tests show that good quality solutions are obtained in a short computational time.

Tirado et al. [25] develop heuristics for the dynamic and stochastic routing problem in industrial shipping. A discrete event simulation reproduces the planning environment in which new cargo requests arrive over time and, whenever a replanning action is scheduled, heuristics are run to produce solutions consistent with currently available information. Three heuristics are developed: (i) a dynamic heuristic that does not use stochastic information, (ii) a multiple scenario approach with consensus based on Bent and Van Henteryck [2] and (iii) a branch-and-regret heuristic based on Hvattum et al. [14]. Both heuristics (ii) and (iii) use an adaptation of the Korsvik et al. [17] tabu search to solve sampled scenarios, but the heuristics differ on the criteria of how to update the current solution which consist of the selection of the best scenario based on a consensus function versus an iterative branch-and-regret procedure to determine the assignment of cargoes to ships. Computational experiments show that the inclusion of stochastic information when solving the routing problem, such as in heuristics (ii) and (iii), produces savings for the shipping company.

3. Problem description

This section presents a concise description of the FSMRTP. For a thorough description and mathematical formulations of the four problem models of Table 1, see Branchini and Armentano [3].

The FSMRTP consists of the deployment of ships to trade routes, the routing of contractual and spot voyages, and the scheduling of the ships. Changes in fleet size and mix are allowed, such as to hire a ship to accommodate a peak of demand or to charter a ship for the remaining of the year if the overall fleet utilization is too low. A solution of the FSMRTP is feasible if all contractual voyages of the planning horizon are served without exceeding ships capacity and violating voyages time window constraints. Spot voyages may be served if feasible and profitable. A solution of the FSMRTP is optimal if it is feasible and if it maximizes overall profit, which is defined as the sum of total revenues (charter, contractual and spot voyages revenues) minus the sum of total costs (fixed, variable, lay-up, hire and operating costs).

3.1. Ships and voyages

Ships have different capacities and may transport different cargo types. Sailing, loading and unloading times may vary among ships and according to the assignment of ships to cargoes. Ships with similar characteristics, such as cargo type, loading/unloading equipment and sailing time, define a ship class.

To solve any of the four problem models, ships must be assigned to voyages. A voyage is defined as a:

a) number of port calls to pick up cargoes;
b) number of port calls to deliver cargoes;
c) fixed quantity of cargo to be picked up and delivered in each port call;
d) time window to start the first port call;
e) time window to finish the last port call which can be based on an estimate of the duration of the voyage.

Each port call has a queue and loading/unloading times. To evaluate if a ship has started service at a port within the time window constraint, the queue time is added to the arrival time to compare with the time window range. Loading and unloading times may vary according to the type of ships and volume of cargoes.

The waiting time of a ship is the time between its arrival and the start of unloading, and it depends on the queue of ships in a port. Even though large waiting times reduce profit, there is not a constraint for the maximum waiting time of a ship in any port call.

The volume of cargo on a ship after each port call cannot exceed the ship capacity. To evaluate these capacity constraints, it is assumed that when a ship reaches a port call, the unloading operation is performed before the loading operation.

A voyage may be also considered as an instance of a trade route. Liner shipping companies make public the set of trade routes to the shipping market and then sell contractual voyages of each trade route. Figure 1 shows an example of a trade route that may load cargo at Itajaí, Santos and Suape ports and then unload cargo at Rotterdam and Hamburg ports.

Figure 1. Example of a trade route

There may be restrictions on the assignment of ships to voyages, for example:

a) Type of ships: some cargoes cannot be assigned to certain type of ships;

b) Port restrictions: draft constraints and requirements of loading/unloading equipment;

c) Contractual obligations: client contracts can determine the type and size of ships that may transport the cargo.

After client contracts have been signed for a specific voyage, other spot voyages with a single port to load cargo and another port to unload cargo, for example, to transport 5.000 tons from Santos to Rotterdam may be accepted by the shipping company if the overall profit is increased and if there is enough capacity at the ship. Spot voyages revenue may represent from 5% to 30% of the total revenues of the shipping company.

3.2. Costs and revenues

The shipping company may operate with own and/or hired ships and may also charter excess ships. A different cost structure occurs in each case.

A fixed cost is incurred if the owner's ship is used during the planning horizon to serve at least one voyage or if an owned ship is chartered. Depending on the charter market, the charter revenue may be smaller or larger than this fixed cost. To charter a ship is similar to renting an asset, in which there is a minimum and maximum renting time, a fixed rent income which contributes to cover the fixed costs, and a variable renting income proportional to the amount of the renting time. The duration of the charter contracts is limited to given lower and upper bounds for the contract period and can be at most as large as the whole planning horizon period. In addition, it is assumed that each ship may be engaged into at most two non-simultaneous charter contracts during the planning horizon. In the case in which more than two contracts are needed to be modeled, the lower and upper bounds for the contract period could be changed to accommodate a larger period of time of multiple contracts.

If the shipping company has no use for some of its ships for a long period of time, an option is to lay-up ships to reduce the fixed cost. Lay-up is to moor a ship at a protected anchorage or berth for a period of time of at least 5-6 months with most onboard systems shut down. This operation decreases fixed and insurance costs, reduces wear and tear of the ship and of the machinery, and may be also combined with maintenance operations. Because of the financial and time commitment of laying-up a ship, the options of laying-up and chartering are considered mutually exclusive for the same ship.

The shipping company may also ships to complement its own fleet. A ship could be hired for a period of time between given lower and upper bounds of the duration of the contract. A fixed cost and a variable cost proportional to the hire period are incurred. It is assumed that only a single hire contract may be settled for each ship during the planning horizon. Similar to the charter assumptions, longer hire periods may be modeled by changing the bounds.

In summary, the costs associated to ships are:

- Fixed costs (\$/year): personnel, supplies, equipment, maintenance, repair, administration (e.g. insurance, office overhead, agency fees), cost of capital (e.g. financing, leasing), make ready to sail costs and all other running costs that do not depend on the distance travelled;
- Lay-up costs (\$/year): administration, cost of capital and lay-up service and maintenance costs;
- Hire fixed costs (\$/contract) and variable costs (\$/day or \$/hour): renting costs to cover administration, cost of capital and profit of the owner of the ship;
- Variable costs:
  - Daily running costs at ports (\$/hours at port): port charges and fuel to maintain ships at port;
  - Fuel costs (\$/nautical mile): fuel for ballast, parcel and full shipload sailing.

Although fuel cost is approximately proportional to the third power of the speed and, sometimes, even to the amount of cargo on the ship, this study considers that ships of distinct classes have different speeds and that the speed of each ship is fixed and given. This seems as a reasonable assumption for strategic and tactical planning problems.

Table 2 summarizes the options that a shipping company has for own and hire ships and the associated costs and revenues of each decision.

Table 2. Costs and revenues associated with each ship mix decision

Does the shipping company own the ship?	Was the ship used to serve voyages?	During the planning horizon, the ship was...	Associated Costs		Associated Revenues
Yes	**Yes**	**used**	•Fixed	•Variable	•Voyages
	No	**chartered**	•Fixed		•Charter fixed and variable
	Yes	**used and chartered**	•Fixed	•Variable	•Voyages •Charter fixed and variable
	No	**laid-up**	Laid-up		-
	No	**not used**	•None[a]		-
No	**Yes**	**hired**	•Hire fixed and variable		•Voyages
	No	**not used**	-		-

[a] *If the company has other fixed costs associated with an unused ship, these costs could be set as lay-up costs.*

4. Tabu search

Tabu search (TS) is a metaheuristic that guides a local search heuristic procedure by using characteristics of the current solution and the history of search in order to explore the solution space beyond local optimality. Short and long term memories are used to store a selective history of the search. In the short term memory, selected attributes that occur in recently visited solutions and tabu activation rules define tabu-active attributes that are stored in a tabu list. Solutions that contain tabu-active attributes are called tabu. This prevents the visit to recent solutions and other solutions that share the tabu-active attributes. This feature prevents cycling and forces the exploration of other regions. The long term memory contains a selective history of complete solutions and attributes of solutions visited during the search. Such elements are used to implement diversification and intensification strategies (Glover and Laguna [13]).

TS is very flexible and there are many ways to define an algorithm for a given problem. We first present the short and long term components of our implementation and the algorithm is presented at the end of this section.

4.1. Objective function, infeasibility and stop criterion

The profit of a solution is evaluated by the calculation of the profit of each route $r \in R$, in which R is the set of ship routes. Thus, the total profit of a solution is defined as $Solution\_Profit = \sum_{r \in R} route\_profit_r$, in which $route\_profit_r$ represents the profit of route $r \in R$.

We allow infeasible solutions relative to time window constraints in order to improve the reachability of regions of feasible solutions which are disconnected or not accessible by simple moves restricted to the feasible region. Therefore, a solution in which a route violates time windows constraints is not discarded and is evaluated by the penalized function $route\_profit_r = profit_r - Pen \times ExceededTime_r$, such that Pen is a non-negative parameter and $ExceededTime_r$ is the sum of all time window violations of route r.

TS stops after a total number of calls of the procedure that calculates $route\_profit_r$ is greater than a predetermined limit parameter (*totalCalcs*). The use of this type of stopping criterion produces a fair comparison among the different variants of the TS algorithm that were implemented. In addition, TS also stops if the pool of solutions (explained in section 4.8) is not updated during 100 iterations.

4.2. Constructive heuristic

The constructive heuristic is inspired by the I1 heuristic of Solomon [24] for the vehicle routing problem with time windows constraints. The goal is to quickly generate a reasonable start solution which may be infeasible with respect to the time windows. The constructive heuristic has three steps:

1. Sort all contractual voyages by ascending order of the latest time of the time window;
2. Sort all vessels by ascending order of net tonnage capacity;

3. Starting from the beginning of the list of step 1, insert each contractual voyage into the best feasible position of the route of the ship with the smallest slack capacity (list of step 2). If there is no feasible insertion, the voyage is inserted into the infeasible position that maximizes overall profit by assigning *Pen*=0.

Step 1 creates a chronological order to select which contractual voyage is inserted first: the earlier a voyage must be served, the earlier this voyage will be selected. Step 2 prioritizes the selection of smaller, and probably cheaper ships, and also saves larger ships to serve larger demands if needed. Finally, step 3 assigns each contractual voyage to a position in a route in which the profit is maximized.

4.3. Neighborhoods

Three neighborhoods are considered: *Insertion*, *Exchange* and *Swap*. *Insertion* consists of all neighboring solutions that may be reached by moving a single contractual voyage from one ship route to another. *Exchange* is equivalent to two subsequent insertion moves, one from a route i to another route j and another insertion from route j to route i. *Swap* is the exchange of *all* voyages of a route i with the voyages of another route j.

The *Swap* neighborhood was implemented with the objective of reaching, with a single move, a neighboring solution in which the routes of a pair of ships are completely exchanged. In other words, *Swap* is an extension of the *Exchange* neighborhood because *Swap* exchanges more than one pair of voyages at a time. However, many times it is not possible to execute the *Swap* move because it is infeasible to assign one or more voyages of one ship to another. For example, suppose that ship v_a serves route r_a and ship v_b serves route r_b. The greater the number of contractual voyages in route r_a (or r_b), the less likely that it is feasible for ship v_b (or v_a) to serve route r_a (or r_b), because of incompatibilities between each swapped voyage and ship capacity, ship location and 'ship x voyage' assignment constraints. Therefore, the $Swap_{(a1,\, a2)}$ variant was also implemented, which indicates that a_1 voyages of the first route are swapped with a_2 voyages of the second route, while the remaining voyages of routes 1 and 2 that differ from those in a_1 and a_2 are inserted into other routes. Thus, in the end, route 1 will only have voyages of route 2, and route 2 will only have voyages of route 1. In addition, in an attempt to reduce the fleet size, the remaining a_1 and a_2 voyages may be inserted into other routes. For example, route 2 would be eliminated if it is feasible and profitable to insert all a_1 voyages into other routes. Figure 2 describes the $Swap_{(a1,\, a2)}$ neighborhood.

$Swap_{(a1,\ a2)}$ was implemented for the following values of the parameters of a_1 and a_2: {(0,2); (0,3); (1,2); (1,3); (2,0); (2,1); (2,2); (2,3); (3,0); (3,1); (3,2); (3,3)}.

Since both neighborhoods *Swap* and $Swap_{(a1,\ a2)}$ are computationally expensive to calculate, especially when the search is in the infeasible region because of a larger number of neighboring solutions, they are disregarded when the search is in the infeasible region.

1. For each consecutive sequence of a_1 voyages of route 1 and consecutive sequence of a_2 voyages of route 2, do:
   1.1. Remove a_1 voyages of route 1;
   1.2. Remove a_2 voyages of route 2;
   1.3. Let *success* = true;
   1.4. For each of the remaining voyages of route 1, do while *success*=true:
      1.4.1. Select a voyage i at random of route 1 and find the best feasible insertion position of i in another route $r \neq 1$ and $r \neq 2$. If a feasible insertion position is found, execute the insertion, otherwise, let *success* = false.
   1.5. For each of the remaining voyages of route 2, do while *success*=true:
      1.5.1. Select a voyage i at random of route 2 and find the best feasible insertion position of i in another route $r \neq 1$ and $r \neq 2$. If a feasible insertion position is found, execute the insertion, otherwise, let *success* = false.
   1.6. If *success*=true (now both routes 1 and 2 are empty):
      1.6.1. Insert a_1 voyages into route 2;
      1.6.2. Insert a_2 voyages into route 1;
      1.6.3. For each voyage of route 1, do (attempt to eliminate route 1):
         1.6.3.1. Select a voyage i at random of route 1 and find the best feasible and profitable insertion position of i in another route $r \neq 1$ and $r \neq 2$. If an insertion position is found, execute the insertion.
      1.6.4. For each voyage of route 2, do (attempt to eliminate route 2):
         1.6.4.1. Select a voyage i at random of route 2 and find the best feasible and profitable insertion position of i in another route $r \neq 1$ and $r \neq 2$. If an insertion position is found, execute the insertion.

Figure 2. $\mathbf{Swap}_{(a1,\,a2)}$ neighborhood

4.4. Candidate list for insertion and exchange moves

A candidate list, as described in Glover and Laguna [13], is applied to the neighborhoods of insertion and exchange moves when the search is either in the feasible or infeasible region. The goal of a candidate list is to restrict the set of neighbor solutions to those which are most promising.

For every insertion and exchange move we compute the value of *delta_distance* = *total_distance_after_move* - *total_distance_before_move*. If *delta_distance* < 0, the move reduces the total sailed distance and, maybe increases profit. On the other hand, a move with a large positive value of *delta_distance* increases sailing costs and is probably not a promising move. Since it is much faster to calculate *delta_distance* than *solution_profit*, it is advantageous to evaluate insertion and exchange moves that lead to low values of *delta_distance*. Therefore, a list of candidates based on a *delta_distance_max* measure is created containing solutions *s* with $delta\_distance_s \leq delta\_distance\_max$.

A parameter $dist\_perc \in (0\%, 100\%]$, which is the percentage of the number of neighboring solutions that must be evaluated, is used to compute the value of *delta_distance_max*. For example, let us suppose that the number of neighbors of a current solution is 10 and that *dist_perc*=60%. In this situation, *delta_distance_max* would be set to the value that results in a total number of 6 neighbors of the current solution. Because *delta_distance_max* is a function of the current set of solution neighbors, *delta_distance_max* varies at each iteration. However, to avoid the computational effort of calculating the exact value of *delta_distance_max* at every iteration, we update the value of *delta_distance_max* each time the search enters or leaves the feasible region as shown next.

4.5. Search in the infeasible region

As discussed in section 4.1, the profit of each route is given by the expression $route\_profit_r = profit_r - Pen \times ExceededTime_r$. If *Pen* equals to a very large positive number, the search is guided to the feasible region. On the other hand, if *Pen* equals to a low positive number, infeasible solutions may be selected if these solutions have high profit. Therefore, the search may be guided either to the feasible region by increasing the value of *Pen* or to the infeasible region by reducing the value of *Pen*.

The search begins with *Pen* equals to a large number to either maintain the search in the feasible region or to find the first feasible solution of the problem when the constructive heuristic is unable to find a feasible solution. Then, the search switches between the feasible and infeasible regions as depicted in Figure 3.

Steps 1 to 4 initialize variables and parameters. The parameter *maxIter* is an estimate of the maximum number of TS iterations, which is proportional to the fleet size and number of contractual voyages (see Table 3). The start values of *PenMin*=5000 and *Pen*=50000 were determined empirically, but they are automatically adjusted in step 7 if these values are too low or high. Step 5 guides the search to the infeasible region and step 6 guides the search to the feasible region. Step 7 is the adaptive adjustment of *Pen* and *PenMin*. If there are too many feasible solutions (*countFeasible* > *totalNumberOfShips* in Step 7.1), *Pen* is divided by 100. Therefore, more infeasible solutions should be evaluated. The variable *countFeasible* is compared with the parameter *totalNumberOfShips* in Step 7.1 because *totalNumberOfShips* is proportional to the problem size. The larger the problem size, the more feasible solutions are evaluated before dividing *Pen* by 100.

It is desirable for the search to stay in the infeasible region just a sufficient number of iterations to guide the search to explore another solution region. In other words, excessive iterations in the infeasible region may be a waste of computing resources. Thus, if there are too many infeasible solutions (Step 7.2), *Pen* is increased by 50% for each infeasible route. For example, if there are 3 infeasible routes in the solution, *Pen* is multiplied by 2.5 $(1.0 + 0.5 \times 3 = 2.5)$. This expression (Step 7.2.2) stimulates the search to move back to the feasible region in a faster pace if there are too many infeasible routes.

1. Let *iterLastSwitch*=0;
2. Let *iterToSwitch*=*maxIter*/10; (*estimate of 10 feasible/infeasible switches during TS run*)
3. Let *PenMin*=5000; *Pen*=50000;
4. Let *infeasibleRegion*=false, *countFeasible*=0; *countInfeasible*=0;

[*Tabu search procedure...*]

5. If *infeasibleRegion*=false AND *currentIter* - *iterLastSwicth* > *itersToSwitch* AND *currentIter* - *iterIncumbentWasFound* > *itersToSwitch* THEN (*goes to the infeasible region*)
    5.1. *infeasibleRegion*=true;
    5.2. *iterLastSwicth = currentIter;*
    5.3. *Pen*=*max*(100, *PenMin*/100);
    5.4. *countFeasible*=*countInfeasible*=0;
6. If *infeasibleRegion*=true AND *currentIter* - *iterLastSwicth* > *itersToSwitch* THEN *(goes to the feasible region)*
    6.1. *infeasibleRegion*=false;
    6.2. *iterLastSwicth = currentIter;*
    6.3. *Pen*=50000;
7. If *infeasibleRegion*= true THEN (*frequently crosses the feasibility border*)
    7.1. If *countFeasible* > *totalNumberOfShips* THEN (*too many feasible solutions, reduces Pen*)
        7.1.1. *Pen* = *max*(100, *Pen*/100);
        7.1.2. *countFeasible*=*countInfeasible*=0;
    7.2. If *countInfeasible* > 10 THEN (*too many infeasible solutions, increases Pen*)
        7.2.1. *PenMin = Pen;* (*stores at PenMin the last value of Pen in the infeasible region*)
        7.2.2. *Pen* = *Pen* x (1.0 + 0.5 x *numberOfInfeasibleRoutes*);
        7.2.3. *countFeasible*=*countInfeasible*=0;

Figure 3. Procedure to guide the search into the feasible or infeasible regions

4.6. Tabu attribute and tenure

The attribute of a solution used in the short term memory is given by the contractual voyages that are assigned to a given ship. Three tabu activation rules are used:

i) if a contractual voyage *i* is moved from ship *a* to ship *b*, voyage *i* must remain on ship *b* for at least t_1 iterations;

ii) if a contractual voyage *i* is moved from ship *a* to ship *b*, voyage *i* must not return to ship *a* for at least t_2 iterations ($t_2 > t_1$);

iii) if an exchange move is executed between two voyages *i* and *j*, both rules (i) and (ii) apply to voyages *i* and *j*;

iv) if a swap move is executed between ships *a* and *b*, another swap move between ships *a* and *b* may only occur after t_3 iterations.

The tabu tenure values of t_1, t_2 and t_3 are randomly selected from a range of [*min*, *max*] values at each executed move. The values of *min* and *max* are related to the test problem size and were determined empirically. Details on the definition of *min* and *max* and all other TS parameters are shown in Table 3 at the end of this section.

Even though *min* and *max* are related to the test problem size, this range of tabu tenure values may not be adequate for the entire search space. Since the number of neighbors in the feasible region is much smaller than the number of neighbors in the infeasible region, the tabu range is adaptively adjusted in both regions by introducing another tabu tenure variable *tabuTenureToAdd* that is a function of the current number of solution neighbors. Then, for example, t_1 is calculated as a random number between [*min*, *max*] plus the value of *tabuTenureToAdd*.

Let *minNumberOfNeighbors* and *maxNumberOfNeighbors* be two parameters proportional to the test problem size and *currentNumberOfNeighbors* be the total number of neighbors of the current iteration. Figure 4 illustrates how *tabuTenureToAdd* is calculated.

Steps 2.1 to 2.3 of Figure 4 reduce the value of *tabuTenureToAdd* by two units in such a manner that the value of *tabuTenureToAdd* is set within the range [-2, -10]. If even with *tabuTenureToAdd*= -10 the search stalls (there are no feasible non-tabu moves to execute), *infeasibleRegion* is automatically set to *true* to increase the number of non-tabu neighbors. On the other hand, Steps 3.1 to 3.3 increase the value of *tabuTenureToAdd* by two units and set the value of *tabuTenureToAdd* within the range [2, 10].

```
1. Let tabuTenureToAdd=0;
                  [Tabu search procedure...]
2. If currentNumberOfNeighbors  < minNumberOfNeighbors THEN
2.1. tabuTenureToAdd = min(tabuTenureToAdd, 0);
2.2. tabuTenureToAdd = tabuTenureToAdd - 2;
2.3. tabuTenureToAdd = max(tabuTenureToAdd, -10);
3. If currentNumberOfNeighbors  > maxNumberOfNeighbors THEN
3.1. tabuTenureToAdd = max(tabuTenureToAdd, 0);
3.2. tabuTenureToAdd = tabuTenureToAdd  + 2;
3.3. tabuTenureToAdd = min(tabuTenureToAdd, 10);
```

Figure 4. Evaluation of variable '*tabuTenureToAdd*'

Finally, the tabu list is emptied, i.e., all neighbors are considered non-tabu each time a new incumbent solution is found. In this case, the search behaves, at least for some iterations, as an ordinary local search procedure, i.e., a simple intensification strategy in the region of the new incumbent solution.

4.7. Selection of neighboring solutions and aspiration criterion

In general, the best neighbor of all neighborhoods (*Insertion*, *Exchange* and *Swap*) is selected at each iteration:

- If the best neighbor is feasible and has a greater profit than the best feasible solution found so far, this best neighbor is selected regardless of the tabu list of section 4.6 (aspiration criterion). In this situation, the incumbent solution is updated;
- Otherwise:
  - If *infeasibleRegion*=true, the non-tabu neighbor with the greatest profit is selected;
  - If *infeasibleRegion*=false, the best non-tabu neighbor is selected considering the following hierarchical objectives: (i) least number of ships and (ii) greatest profit.

If *infeasibleRegion*=true, it is expected that the search visits high profit solutions, which may or may not have the same number of ships. Many times, the number of ships is reduced. On the other hand, if *infeasibleRegion*=false, there are two hierarchical objectives. Objective (i) is introduced because the search space of solutions with fewer ships produces, normally, solutions with higher profits. Thus, it is expected that the search first reduces the number of ships and then seeks solutions of high profit.

4.8. Pool of solutions and distance among solutions

High quality and diverse solutions found during the search are stored in a *pool*. The solution pool has two main objectives, namely, to give to the planner of the shipping company more alternatives of high quality solutions to be implemented, and to use the information of high quality solutions to find other high quality solutions.

Up to two solutions pools are managed during the search: the first pool contains solutions without spot voyages and the second pool contains solutions with spot voyages. The first pool (without spot voyages) is used by the tabu search components of sections 4.3 to 4.8 for the four problem models. These components are computationally intensive and they were designed to effectively address the set of contractual voyages. The second pool (with spot voyages) is used for the SIMPLE.CvSv and FULL.CvSv problem models and by the components 'Use of pool information to guide the search' and 'Greedy insertion heuristic for spot voyages', which are explained after this section.

Both pools have a maximum number of *poolMax* solutions and a solution is inserted into the pool only if it is different from all other solutions that are already in the pool. To maintain a minimum diversity among the solutions of the pool, two variants of pool management were implemented, a pool based on fleet composition and a pool based on distance among the solutions.

a) Pool management based on fleet composition (PMF):

A new solution is inserted into the pool if it is better than the worst solution of the pool or the pool has less than *poolMax* solutions. In addition, either one of the following statements must occur:

- The set of ships of the new solution is different from the set of ships of all other solutions in the pool; in this case the new solution replaces the worst solution;
- The set of ships of the new solution is equal to the set of ships of a solution $s_i \in$pool and the profit of the new solution is greater than the profit of s_i; in this case the new solution replaces s_i.

Therefore, PMF focuses on storing high quality solutions with different fleets. This is useful because it gives the planner a set of *poolMax* solutions with the greatest profit found for each fleet composition.

b) Pool management based on distance (PMD):

This pool stores a new solution according to a measure of distance among solutions that considers the attributes of fleet composition and assignment of contractual voyages to ships[4]. The distance $dist_{(s1,s2)}$ between two solutions s_1 and s_2 if given by

$$dist_{(s_1,s_2)} = \frac{numberDiffCvAssignments + numberDiffShips \times avgNumberCvsPerShip}{|CV|}$$

in which:

numberDiffCvAssignments = the total number of contractual voyages of solution s_1 that are assigned to a different ship in solution s_2;

numberDiffShips = the total number of ships used in s_1 that is not used in s_2, plus the total number of ships used in s_2 that is not used in s_1;

CV = the set of contractual voyages;

$$avgNumberCvsPerShip = max\left(\frac{|CV|}{numberOfUsedShips_{s_1}}, \frac{|CV|}{numberOfUsedShips_{s_2}}\right).$$

This measure is symmetrical, $dist_{(s1,s2)}=dist_{(s2,s1)}$, and if solution s_1 is equal to s_2, $dist_{(s1,s2)}=0$. The more different the assignment of contractual voyages are between solutions s_1 and s_2, the greater the value of $dist_{(s1,s2)}$. In addition, different fleet compositions are included in the measure with a weight proportional to the average number of contractual voyages per ship. Because of this inclusion, $dist_{(s1,s2)}$ may assume values greater than 1.

A new solution *s* is inserted into the pool if it satisfies one of the following criteria, testing each criterion according to the order presented next:

1. If the number of solutions of the pool is less than *poolMax*, *s* is inserted into the pool if *s* is better than the incumbent solution s_{best} or if *s* has at least a distance measure of 0.1 among all other solutions of the pool, $dist_{(s,pool)} = min\left(dist_{(s,s_i)} : \forall s_i \in pool\right) \geq 0.1$. The parameter 0.1 was determined empirically after testing values within the range [0.05, 0.30].

2. If the pool has *poolMax* solutions, *s* is inserted into the pool replacing the worst solution s_w of the pool if the profit of *s* is greater than the profit of s_w and if the distance measure between *s* and all other solutions, except s_w is greater than or equal to 0.1, $min\left(dist_{(s,s_i)} : \forall s_i \in pool, s_i \neq s_w\right) \geq 0.1$, and if the percentage difference between the profit of s_w

[4] Other types of distance measures may be applied, such as a distance measure that also considers the sequence of voyages in a route (e.g. the information about sequence of voyages in a route is used in the next tabu component of section 4.9). Because one of the main goals of TS is fleet deployment, a distance measure focused on fleet composition and assignment of contractual voyages to ships is used.

and the profit of s_{best} is greater than a percentage parameter threshold, such as $\left(1.0-\frac{Profit_{s_w}}{Profit_{s_{best}}}\right)\times 100 \geq 4\%$. The parameter 4% guides the profit range of the solutions of the pool so that the profit difference between the solution with the highest quality (s_{best}) and the solution with the lowest quality (s_w) is around 4%.

3. If the pool has *poolMax* solutions and criterion (2) is not met, *s* is inserted into the pool replacing a solution $s_p \in$pool, if either of the following conditions (3.1) or (3.2) occurs:

3.1. *s* is better than the best solution of the pool. In this case, *s* replaces a solution $s_p \in$pool in the following order:

3.1.1. if $min\left(dist_{(s,s_i)} : \forall s_i \in pool, s_i \neq s_p\right) \geq 0.1$ and the replacement of s_p by *s* either increases the maximum distance $distMax = max\left(dist_{(s_i,pool)} = min\left(dist_{(s_i,s_j)} : \forall s_j \in pool, s_i \neq s_j\right) \forall s_i \in pool\right)$, or, maintains the maximum distance but increases the average distance $distAvg = \frac{\sum_{s_i,s_j \in pool i \neq j} dist_{(s_i,s_j)}}{poolMax \times (poolMax\text{-}1)}$, *s* replaces s_p;

3.1.2. if (3.1.1) does not hold, *s* replaces the solution s_p that is closest to *s*. If there is a tie among two or more solutions that belong to the set of the closest solutions to *s*, the solution with the worst profit of this set is replaced.

3.2. *s* replaces a solution $s_p \in$pool if *s* is better than the worst solution of the pool and $min\left(dist_{(s,s_i)} : \forall s_i \in pool, s_i \neq s_p\right) \geq 0.1$ and:

3.2.1. *s* increases the maximum distance among the solutions of the pool, or,

3.2.2. *s* maintains the maximum distance but increases the average distance of the pool, or,

3.2.3. *s* maintains the maximum and average distance and has greater profit than s_p.

Thus, criterion (1) creates a pool of *poolMax* solutions respecting, whenever possible, the minimum distance measure of 0.1 among the solutions. After the pool has *poolMax* solutions, criteria (2) and (3) may update the solutions of the pool whenever a solution *s* with profit larger than that of s_w is found. If the percentage difference between the profits of s_w and s_{best} is greater than 4% and all other criteria of (2) hold, *s* replace s_w. Otherwise, criterion (3.1) updates the pool whenever the profit of *s* is better than the profit of s_{best} and criterion (3.2) uses the measures *distMax* and *distAvg* to test if *s* may replace a solution s_p of the pool considering three goals: (i) to have solutions that are very diverse (maximize *distMax*), (ii) to maximize the average diversification (maximize *distAvg*) and, (iii) to improve the solutions quality (maximize profit).

Simpler rules for pool management were implemented and tested, but it was the set of the previous rules that produced, at the end, a pool with an adequate balance between quality and diversification. For instance, the following problems were encountered when only simpler rules, such as the sole maximization of the maximum distance, were used:

- If TS finds, for example, three new incumbent solutions in a sequence or within a short number of iterations, very common in local search based methods such as TS, these three solutions are probably similar to each other and would be inserted into the pool, also in a sequence, reducing the pool diversity. This problem is solved with the inclusion of the ordering rules (3.1.1) and, in particular, (3.1.2) for selecting which solution s_p would leave the pool;
- Once a very diverse solution is found and inserted into the pool, the pool is seldom updated because the search rarely finds other solutions that increase the maximum distance and, at the same time, are better than the worst solution of the pool. In this case, the pool is usually updated when TS finds a new incumbent solution. Therefore, the resulting pool often contains two groups of solutions, one group with many solutions similar to the incumbent solution and another group with few diverse solutions. This problem was solved with the inclusion of criteria (2), (3.1.1), (3.2.2) and (3.2.3).

4.9. Use of pool information to guide the search

After several insertions of solutions into the pool, the pool contains *poolMax* high quality and diverse solutions. The information of these solutions may be combined to guide the search to find new and, hopefully, better solutions.

When the component '4.9. Use of pool information to guide the search' is activated, the information of the pool without spot voyages (for the SIMPLE.Cv and FULL.Cv problem models) and of the pool with spot voyages (for the SIMPLE.CvSv and FULL.CvSv problem models) are used to prioritize moves that approximate the structure of the current solution to that of the solutions of the pool.

To gather information about the structure of the solutions of the pool, the vector $voyageIsTheFirstOfRoute_{(i)}$, of $|CV|$ positions, is used to store the number of times each voyage $i \in CV$ is the first voyage of a route in any solution of the pool. The matrix

$voyage2AfterVoyage1_{(i,j)}$, of $|CV|$ by $|CV|$ positions, stores the number of times voyage $j \in CV$ is served immediately after voyage $i \in CV$ in any solution of the pool. In addition, let $maxFrequency = max(voyageIsTheFirstOfRoute_{(i)}, voyage2AfterVoyage1_{(i,j)}), \forall i, j \in CV, i \neq j,$ and let $frequency_r$ equals to the sum of the corresponding values of $voyageIsTheFirstOfRoute_{(i)}$ and $voyage2AfterVoyage1_{(i,j)}$ of a given route r. Also, let *maxPositiveProfitIncrease* be the maximum non-negative profit increase between two iterations during the past 20 iterations.

Finally, to use the information extracted from the pool to bias the search, the $route\_profit_r$ expression is modified as follows:

$$route\_profit_r = profit_r - Pen \times ExceededTime_r + maxPositiveProfitIncrease \times \frac{frequency_r}{maxFrequency}.$$

Hence, the more similar the sequence of voyages of a route r is to the sequence of voyages of the routes of the pool, the greater is the value of the expression $route\_profit_r$.

4.10. Greedy insertion heuristic for spot voyages

A greedy insertion heuristic was implemented to insert spot voyages before, during and after contractual voyages. This heuristic is executed when the pool of solutions is updated and, also, may be executed at each iteration.

This is a simple and fast heuristic that, given a set of routes of contractual voyages, inserts the greatest number of spot voyages into routes to increase overall profit.

The first step is to calculate the profit increase of the best feasible insertion positions of the loading and unloading port calls of all spot voyages. Then, the insertion that produces the largest profit increase is executed, and the set of the next possible insertions is updated. This process is repeated while it is profitable to insert spot voyages.

4.11. Tabu Search variants

Figure 5 shows the framework associated with the implementation of TS that consists of an orderly combination its components.

Phase 1 uses all tabu components except for ‘4.9. Use of pool information to guide the search’. The component ‘4.10. Greedy insertion heuristic for spot voyages’ is used only when a solution is inserted into the pool. The objective of Phase 1 is to find the first set of feasible solutions.

		Results
Phase 1:	*Initialization*: Constructive heuristic and TS. Greedy insertion of SVs only when pool is updated[a].	First pool of solutions
Phase 2:	*Diversification (tree search)*: Forbids the use of least profitable ships during constructive heuristic and TS. Greedy insertion of SVs only when pool is updated[a].	Diverse solutions inserted into pool
Phase 3:	*Improvement*: Restart from solutions of the pool (all ships are used). Greedy insertion of SVs at every iteration[a].	Improvement of pool solutions
Phase 4:	*Intensification*: Restart from solutions of the pool. Use of pool information. Greedy insertion of SVs at every iteration[a].	Improvement of pool solutions

[a]*Applicable if problem type is SIMPLE.CvSv or FULL.CvSv.*

Figure 5. TS framework

Phase 2 is a diversification procedure that consists of a restricted tree search, which resembles the filtered beam search heuristic developed by Ow and Morton [21]. This phase starts from the best solution of Phase 1 and sorts the routes with their respective ships in ascending order of profit, so that the ships at the top of the list are those with the least profit that are used to control which ships are available for the constructive heuristic and TS. Then, each of the top 5 ships in the list is considered to be forbidden for the constructive and TS heuristics. This corresponds to the branching of the root node into 5 nodes of a tree, each node with a distinct forbidden ship. If one of the five runs of the TS produces a solution with greater profit than the incumbent obtained in the root node, the process is repeated, forbidding the use of the ship of the first level tree and the next 5 ships of the current list, yielding 5 nodes in the second level tree. The search is stopped if it is not possible to find better solutions from all tree nodes. In addition, to avoid excessive search time in similar nodes of the tree, each ship may be forbidden at most 3 times. Each TS run of Phase 2 is executed for approximately 400 iterations and the result of this phase is a pool of diverse solutions regarding fleet composition and also distance measure if PMD is applied.

Phase 3 improves each solution of the pool by the restart of TS associated with the cleaning of all short term memory. If applicable, the component ‘4.10. Greedy insertion heuristic for spot voyages' is employed at each iteration of TS. First, pool solutions are sorted by descending order of profit, and starting from the top solution of the pool, TS is executed for about 700 iterations. If a new solution is found and inserted into the pool, this solution is also a candidate for the restarting procedure. After TS is finished, the next solution of the pool list is used for the restarting process. Phase 3 stops when there are no new solutions to restart with or when the number of restarts equals to the parameter *phase3_max_restarts*. The result of this phase is the quality improvement of pool solutions.

Phase 4 is very similar to Phase 3. The difference is that the component '4.9. Use of pool information to guide the search' is activated in Phase 4. This phase also attempts to further improve the quality of pool solutions for at most *phase4_max_restarts*.

The parameters of TS are shown in Table 3. Most parameters are proportional to the size of the problem, which is the product of the cardinality of the set V of ships and the set CV of contractual voyages. The expressions and values of Table 3 are the result of computational experiments dedicated to the parameter setting of the tabu search. These tests were executed on a reduced set of 24 test problems extracted from the complete set of 56 test problems presented in the next section.

The first four parameters of Table 3 limit the total number of iterations of each phase of TS. First, the parameter *maxIter* is calculated accordingly to the upper bound of each TS phase and then *maxIter* is used to compute the parameter *totalCalcs* for the TS stop criterion. *PoolMax* determines the size of the pool of solutions (either 5 or 10 solutions). The next two parameters define the minimum and maximum number of neighboring solutions to update the value of the variable *tabuTenureToAdd* (Figure 4). The last three parameters calculate the tabu tenure values t_1, t_2 and t_3 described in section 4.6. The expressions ti_{LB} and ti_{UB}, $i \in \{1,2,3\}$, refer to the lower and upper bounds of parameter t_i, $i \in \{1,2,3\}$.

Table 3. TS parameters

TS Parameter	Description	Expression	Lower bound (LB)	Upper bound (UB)
maxIter (phase 1)	Estimate of the maximum number of iterations of TS	$\left\lfloor 40 \times \sqrt{\lvert V \rvert \times \lvert CV \rvert} \right\rfloor$	0	2000
maxIter (phase 2)	Estimate of the maximum number of iterations of TS	$\left\lfloor 40 \times \sqrt{\lvert V \rvert \times \lvert CV \rvert} \right\rfloor$	0	400
maxIter (phases 3 and 4)	Estimate of the maximum number of iterations of TS	$\left\lfloor 40 \times \sqrt{\lvert V \rvert \times \lvert CV \rvert} \right\rfloor$	0	700
totalCalcs	Stop criterion: maximum number of calls to the procedure that calculates *route_profit*	$\left\lfloor 6 \times maxIter \times \lvert V \rvert \times \lvert CV \rvert \right\rfloor$	0	∞
poolMax	Maximum number of solutions in the pool	na	5	10
minNumber OfNeighbors	If the number of neighboring solutions is less than this value, *tabuTenureToAdd* is reduced	$\left\lfloor 8 \times \sqrt{\lvert V \rvert \times \lvert CV \rvert} \right\rfloor$	0	∞
maxNumber OfNeighbors	If the number of neighboring solutions is greater than this value, *tabuTenureToAdd* is increased	$4 \times minNumberOfNeighbors$	0	∞
tabu tenure t_1	If a contractual voyage i is moved from ship a to ship b, voyage i must remain on ship b for at least t_1 iterations	$max\begin{pmatrix} \frac{t1_{LB}}{2}, random \in (t1_{LB}, t1_{UB}) + \\ tabuTenureToAdd \end{pmatrix}$	$\left\lfloor \frac{\sqrt{\lvert V \rvert \times \lvert CV \rvert}}{3} \right\rfloor$ and bounded by [5, 12]	$max\begin{pmatrix} \lfloor 1.5 \times t1_{LB} \rfloor, \\ t1_{LB} + 5 \end{pmatrix}$
tabu tenure t_2	If a contractual voyage i is moved from ship a to ship b, voyage i must not return to ship a for at least t_2 iterations	$max\begin{pmatrix} \frac{t2_{LB}}{2}, random \in (t2_{LB}, t2_{UB}) + \\ tabuTenureToAdd \end{pmatrix}$	$\left\lfloor \frac{t1_{LB} + t1_{UB}}{2} \right\rfloor$	$\lfloor 1.5 \times t2_{LB} \rfloor$
tabu tenure t_3	If a swap move is executed between ships a and b, another swap move between ships a and b may only occur after t_3 iterations	$max\begin{pmatrix} \frac{t3_{LB}}{2}, random \in (t3_{LB}, t3_{UB}) + \\ tabuTenureToAdd \end{pmatrix}$	$t2_{UB}$	$\lfloor 1.5 \times t3_{LB} \rfloor$

Finally, six variants of the TS algorithm were implemented considering the tabu components and phases as shown in Table 4.

Table 4. Implementations of TS variants

TS variation	List of candidates?	Pool management	*phase3_ max_restarts*	Apply Phase 4? (*phase4_max_restarts*)
TSNone	No	PMF	4 x *poolMax*	no
TSPMF	No	PMF	2 x *poolMax*	yes, (2 x *poolMax*)
TSPMD	No	PMD	2 x *poolMax*	yes, (2 x *poolMax*)
TSlcNone	yes[a]	PMF	4 x *poolMax*	no
TSlcPMF	yes[a]	PMF	2 x *poolMax*	yes, (2 x *poolMax*)
TSlcPMD	yes[a]	PMD	2 x *poolMax*	yes, (2 x *poolMax*)

[a]*With dist_perc set to 70% for the feasible region and to 50% for the infeasible region.*

The TS variations with ‘None’ have no Phase 4, but the Phase 3 of these variations has at most 4 *x* *poolMax* restarts. The ‘PMF’ and ‘PMD’ variations have Phase 4 with, respectively, PMF and PMD pool management. All variations with ‘lc’ make use of the list of candidates for the insertion and exchange neighborhoods.

The six variants of TS (Table 4) were tested with *poolMax*= 5 and *poolMax*=10, resulting in a total of 12 TS tests: TSNone5, TSPMF5, TSPMD5, TSlcNone5, TSlcPMF5, TSlcPMD5 and TSNone10, TSPMF10, TSPMD10, TSlcNone10, TSlcPMF10, TSlcPMD10.

5. Computational study

The TS variants of Table 4 were implemented in C++ and computational tests were executed in an Intel Xeon 2.83 GHz, 8 GB RAM computer with the Ubuntu operating system. The results of the TS variants are compared to those obtained in Branchini and Armentano [3], which used CPLEX 12.4 to solve the related MIP problems of the same data set.

5.1. Set of test problems

A random test problem generator that considers real world assumptions and parameters was developed to create a set of 56 test problems, 14 test problems for each one of the 4 models of Table 1. Although the test problems do not reflect a specific liner operation, the parameters of each problem were defined within a range of values that could represent practical ship routing operations.

The number of ships of a company ranges from 18 to 50 ships plus 6 to 32 hire ships, resulting in a total of 24 to 82 ships. Each ship belongs to a ship class named Handysize (two types), Supramax, Panamax and Capesize (also two types), which determine most of ships parameters such as capacity, sailing speed and costs.

A set of 53 worldwide ports is used to create the set of demands. A distance matrix among ports was created based on ports latitude and longitude. In addition, for each pair ‘port *x* ship class’, a service time, loading/unloading rate and running costs were determined randomly within real world assumptions. Also, the location of the ship at the time it is available is a port selected randomly from the set of 53 ports.

Each problem of the data set has between 30 and 110 contractual voyages and between 10 and 34 spot voyages, which corresponds to a ratio of contractual and spot voyages around 75%. The amount of cargo at each port call, the voyages’ time windows, duration and revenue are randomly selected.

Contractual voyages were constructed based on a set of 12 trade routes with a number of port calls between 15 and 30. Spot voyages have a single port call to load and a single port call to unload the cargo. The origin and destination port calls are chosen randomly from the set of 53 ports.

5.2. Computational results

This section presents the summary of the computational results. For the complete results tables of the TS variants and a detailed discussion of the computational experiments refer to Appendix A. Moreover, Appendix B presents the results of the constructive heuristic.

Table 5 compares the performance of the 12 variants of TS considering the 4 problem models of Table 1. The results of the three MIP based solution methods of Branchini and Armentano [3] are also included in Table 5. These three MIP based methods are:

1) MIP Exact (MIPE): CPLEX implementation of the mixed integer programming model that is defined on a directed graph whose nodes represent contractual and spot voyages. All feasible ship routes are constructed through sets of arcs of this graph. If CPLEX is able to find the optimal solution of the MIPE model during the computational time of 24 hours, this optimal solution is the best feasible solution of the problem.

2) MIP Best Nodes without the nodes that combine contractual and spot voyages into a single node (MIPBN00): CPLEX implementation of the mixed integer programming model of a reduced graph that does not represent all feasible solutions of the original problem. Thus, there is no guarantee that the best feasible solution found by CPLEX for MIPBN00 is the optimal solution of the problem.

3) MIP Best Nodes with some nodes that combine contractual and spot voyages (MIPBN15): CPLEX implementation of the mixed integer programming model of a reduced graph in which there are some[5] nodes that combine contractual and spot voyages into a single node. The number of nodes of the graph optimized in the MIPBN15 is larger than that of MIPBN00, however, the graph still does not represent all feasible solutions of the original problem. Hence, there is no guarantee that the best feasible solution found by CPLEX for MIPBN15 is the optimal solution of the problem.

[5] The label '15' of MIPBN15 means that the best (top '1') node of each triplet "contractual voyage *x* subsets of spot voyages *x* ship" is selected and that, afterwards, the top '5' nodes of each pair "contractual voyage *x* ship" are introduced into the graph (details in Branchini and Armentano [3]).

Table 5 presents six rows for each problem model (SIMPLE.Cv, SIMPLE.CvSv, FULL.Cv and FULL.CvSv). The first row contains the number of best solutions (solutions with the largest profit) that each has found considering the 14 test problems of each problem model. The second row contains the number of optimal solutions found by each method. A solution is only considered to be optimal if its profit is equal to the profit of the best feasible solution found by MIPE that was proved to be optimal by CPLEX. The next three rows show, respectively, the minimum, average and maximum percentage between the profit found by each method and the best profit when considering the results of all fifteen columns. The greater the percentage value, the better the performance of the method (maximum difference is 0.0%). The fifth row shows the average computational time, in seconds, that each method required to solve one test problem of each problem model. The MIP methods and TS variants that produced the best results for each problem model are highlighted with a gray background.

The following conclusions are obtained from the computational results of Table 5:

- MIPE solved to optimality all 14 SIMPLE.Cv test problems in a short computational time;
- MIPBN15 and TSPMD10 are the best average performers for the SIMPLE.CvSv test problems. The average computational time of TSPMD10 is about 87% lower than that of MIPBN15;
- FULL.Cv test problems are also solved by MIPE. However, the average computational time for large problems is also high. If high quality feasible solutions are required in a shorter computational time, TSlcPMD10, or even TSlcPMD5 and TSPMD5, should be applied;
- Only the smallest FULL.CvSv test problems may be solved by MIP based methods. MIPBN00 was selected as the best MIP based method for the FULL.CvSv because it was the only MIP based method that was able to find feasible solutions for 13 out of the 14 test problems (MIPE and MIPBN15 were only able to find feasible solutions for 10 test problems). In general, TSlcPMD5 and TSlcPMD10 have the smallest average difference from the best solutions and are the best choice for these problems;
- The average computational time of TS variants ranges from 20 minutes (TSNone5) to 3.7 hours (TSlcPMD10). On the other hand, MIP based methods have a much larger computational time. Except for the SIMPLE.Cv test problems, the average computational time for the MIP based methods ranges from 7 hours to 17 hours.

Table 5. Summary of the computational results

			MIPE	MIPBN00	MIPBN15	TSNone5	TSNone10	TSIcNone5	TSIcNone10	TSPMF5	TSPMF10	TSIcPMF5	TSIcPMF10	TSPMD5	TSPMD10	TSIcPMD5	TSIcPMD10
Simple.Cv	# of best solutions		14	na	na	3	4	3	3	3	4	3	3	4	4	3	4
	# of optimal solutions		14	na	na	3	4	3	3	3	4	3	3	4	4	3	4
	Δ from best solution	Min.	0.0%	na	na	-5.8%	-4.7%	-4.9%	-4.9%	-4.7%	-3.7%	-4.9%	-4.9%	-4.6%	-3.8%	-4.0%	-3.7%
		Avg.	0.0%	na	na	-1.4%	-1.2%	-1.5%	-1.4%	-1.1%	-1.0%	-1.2%	-1.4%	-1.1%	-1.0%	-1.0%	-0.9%
		Max.	0.0%	na	na	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
	Avg. time (sec)		105	na	na	1,122	1,636	1,277	1,593	1,729	2,678	1,874	2,621	1,718	3,324	1,913	3,762
Simple.CvSv	# of best solutions		7[a]	1	7	0	2	0	0	2	2	0	0	2	2	1	2
	# of optimal solutions		6	0	2	0	2	0	0	1	2	0	0	2	2	1	2
	Δ from best solution	Min.	-4.6%	-16.6%	-2.0%	-3.6%	-4.0%	-3.6%	-4.2%	-3.6%	-3.7%	-3.6%	-4.2%	-4.1%	-3.8%	-4.1%	-4.3%
		Avg.	-0.7%	-3.8%	-0.3%	-1.2%	-1.4%	-1.4%	-1.5%	-1.2%	-1.3%	-1.3%	-1.4%	-1.1%	-0.9%	-1.6%	-1.3%
		Max.	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
	Avg. time (sec)		49,686	30,663	42,351	2,292	3,994	2,043	3,203	2,614	4,876	2,563	4,578	2,850	5,407	2,846	5,322
Full.Cv	# of best solutions		14	na	na	0	0	0	0	0	0	0	0	0	0	0	0
	# of optimal solutions		11	na	na	0	0	0	0	0	0	0	0	0	0	0	0
	Δ from best solution	Min.	0.0%	na	na	-3.2%	-3.2%	-2.3%	-2.3%	-3.2%	-3.2%	-2.3%	-2.3%	-2.2%	-3.2%	-2.3%	-2.1%
		Avg.	0.0%	na	na	-1.5%	-1.4%	-1.5%	-1.3%	-1.4%	-1.4%	-1.4%	-1.3%	-1.2%	-1.2%	-1.3%	-1.1%
		Max.	0.0%	na	na	-0.2%	-0.2%	-0.5%	-0.3%	-0.2%	-0.2%	-0.5%	-0.3%	-0.2%	-0.1%	-0.1%	-0.1%
	Avg. time (sec)		25,252	na	na	2,959	5,669	3,406	5,823	4,855	9,170	5,050	9,301	6,794	14,727	6,889	13,271
Full.CvSv	# of best solutions		3[b]	1[c]	3[b]	0	0	2	2	0	1	2	3	0	3	2	4
	# of optimal solutions		3	0	2	0	0	0	0	0	0	0	0	0	0	0	0
	Δ from best solution	Min.	-21.9%	-21.8%	-10.0%	-2.1%	-2.4%	-2.2%	-2.1%	-2.1%	-2.4%	-2.1%	-2.1%	-1.9%	-2.4%	-2.0%	-1.8%
		Avg.	-7.0%	-5.8%	-4.1%	-0.7%	-0.8%	-0.9%	-0.8%	-0.7%	-0.7%	-0.8%	-0.8%	-0.7%	-0.7%	-0.6%	-0.6%
		Max.	0.0%	0.0%	0.0%	-0.1%	-0.1%	0.0%	0.0%	-0.1%	0.0%	0.0%	0.0%	-0.1%	0.0%	0.0%	0.0%
	Avg. time (sec)		70,716	61,423	69,161	2,863	4,970	3,193	4,411	4,247	7,634	4,262	7,219	5,243	9,590	4,782	9,516

[a]*MIPE was unable to find a feasible solution for 3 of the 14 SIMPLE.CvSv test problems.*
[b]*MIPE and MIPBN15 were unable to find a feasible solution for 4 of the 14 FULL.CvSv test problems.*
[c]*MIPBN00 did not find a feasible solution for one FULL.CvSv test problem.*
na: the MIPBN00 and MIPBN15 methods are not applied to the SIMPLE.Cv and FULL.Cv problem models, since these models do not have spot voyages.

Further conclusions are drawn from the results of Table 5 and also the discussion of Appendix A:

- In general, the results of the TS variants with *poolMax*=10 are better than the results of TS variants with *poolMax*=5. Also, the computational time of the TS variants with *poolMax*=10 is about twice as large as the computational time of TS variants with *poolMax*=5;
- There is not a significant difference between the quality of the solutions found by TS variants with candidate list, and that of TS variants without candidate list;
- The pool solutions of the TS variants with pool management based on distance (PMD) are more diverse than that of the TS variants with pool management based on fleet composition (PMF);

- Phase 4 significantly increases the total computational time, and it yields a marginal improvement in the quality of the solutions.

6. Conclusions and future work

This research presented a tabu search to tackle planning problems faced by liner shipping companies in maritime logistics. Computational tests were executed on a proposed set of 56 test problems that were based on real world data. Although exact methods, such as MIPE, were able to solve small to medium problems, the ability to obtain feasible and high quality solutions with such methods is reduced as problem size increases. In this case, implementations of variants of tabu search showed more adequate.

A constructive heuristic and twelve variants of tabu search were proposed. These variants make use of insertion, exchange and swap neighborhoods, and infeasible regions are explored by the insertion and exchange neighborhoods. A short term memory, based on three tabu tenure policies, was implemented to avoid cycling and a long term memory, based on the frequency of attributes found in a pool of high quality and diverse solutions, was used to further improve the search. A pool of high quality solutions also adds flexibility for the planner of the shipping company to choose which plan should be put into practice.

The application of pool information to guide the search (Phase 4) further enhanced solution quality and the use of a distance measure proved to be a more efficient to manage the pool of solutions. In addition, pools with 10 solutions generated better results than pools with 5 solutions.

Finally, the following topics of future work related to this research are presented.

- Build a TS neighborhood specifically for spot voyages. A custom build TS neighborhood for spot voyages would probably yield better results than the greedy heuristic of section 4.10. The challenge is to add this neighborhood without requiring a large computational effort.
- Implement a parallel computing version of TS. Since TS is based on creating and maintaining a pool of high quality and diverse solutions, different variants (and even variants with different parameters, e.g. tabu tenure) could be executed simultaneously on several CPU processors. Every time a new solution is found by a process, the same pool management criteria would be applied. Parallelization could be implemented from the beginning of the method and throughout the 4 Phases of TS. This could drastically reduce the computational time of TS.

- Apply other solution recombination and improvement procedures, such as path relinking (Glover and Laguna [13]), to all solutions of the pool of TS. In this case, other types of distance measures, such as a measure that also considers the sequence of voyages in routes, may be used to control the diversity of the solutions of the pool.
- Modify the model and TS to optimize a set of spot voyages scenarios instead of a single scenario. This multiple scenario approach yields solutions that are more robust, especially when the information regarding voyages revenue, set of ports and cargo are highly uncertain.

References

[1] Álvarez JF. Joint routing and deployment of a fleet of container vessels. Maritime Economics & Logistics 2009;11:186-208.

[2] Bent RW, Van Hentenryck P. Scenario-based planning for partially dynamic vehicle routing with stochastic customers. Operations Research 2004;52:977-987.

[3] Branchini RM, Armentano VA. Fleet deployment optimization in liner shipping with spot voyages. European Journal of Operational Research; unpublished results.

[4] Brønmo G, Christiansen M, Fagerholt K, Nygreen B. A multi-start local search heuristic for ship scheduling – a computational study. Computers and Operations Research 2007;34(3):900-917.

[5] Brønmo G, Christiansen M, Nygreen B. Ship routing and scheduling with flexible cargo sizes. Journal of the Operational Research Society 2007;58:1167-1177.

[6] Brønmo G, Nygreen B, Lysgaard J. Column generation approaches to ship scheduling with flexible cargo sizes. European Journal of Operational Research 2010;200(3):139-150.

[7] Christiansen M, Fagerholt K, Ronen D. Ship routing and scheduling – status and perspectives. Transportation Science 2004;38(1):1-18.

[8] Christiansen M, Fagerholt K, Nygreen B, Ronen D. Handbooks in Operations Research and Management Science: Transportation. In: Barnhart C, Laporte G, editors. Maritime Transportation, Amsterdam: North-Holland; 2007, p. 189-284.

[9] Christiansen M, Fagerholt K, Flatberg T, Haugen Ø, Kloster O, Lund EH. Maritime inventory routing with multiple products: A case study from the cement market. European Journal of Operational Research 2011;208:86-94.

[10] Christiansen M, Fagerholt K, Nygreen B, Ronen D. Ship Routing and Scheduling in the New Millennium. European Journal of Operational Research 2013;228(3):467-483.

[11] Cordeau J-F, Laporte G, Mercier A. A unified tabu search heuristic for vehicle routing problems with time windows. Journal of the Operational Research Society 2001;52:928-936.

[12] Fagerholt K, Johnsen TAV, Lindstad H. Fleet deployment in liner shipping–a case study. Maritime Policy & Management 2009;36(5):397-409.

[13] Glover F, Laguna M. Tabu Search. Kluwer Academic Publishers; 1997.

[14] Hvattum LM, Løkketangen A, Laporte G. A branch-and-regret heuristic for stochastic and dynamic vehicle routing problems. Networks 2007;49:330-340.

[15] Karlaftis MG, Kepaptsoglou K, Sambracos E. Containership routing with time deadlines and simultaneous deliveries and pick-ups. Transportation Research Part E 2009;45:210-221.

[16] Korsvik JE, Fagerholt K. A tabu search heuristic for ship routing and scheduling with flexible cargo quantities. Journal of Heuristics 2010;16(2):117-137.

[17] Korsvik JE, Fagerholt K, Laporte G. A tabu search heuristic for ship routing and scheduling. Journal of the Operational Research Society 2010;61:594-603.

[18] Korsvik JE, Fagerholt K, Laporte G. A large neighbourhood search heuristic for ship routing and scheduling with split loads. Computers & Operations Research 2011;38:474-483.

[19] Lawrence S A. International sea transport: the years ahead. Lexington Books, Lexington MA; 1972.

[20] Norstad I, Fagerholt K, Laporte G. Tramp ship routing and scheduling with speed optimization. Transportation Research Part C 2011;19:853-865.

[21] Ow PS, Morton TE. Filtered beam search in scheduling. International Journal of Production Research 1988;26:297-307.

[22] Ronen D. Cargo ships routing and scheduling: Survey of models and problem. European Journal of Operational Research 1983;12:119-126.

[23] Ronen D. Ship scheduling: The last decade. European Journal of Operational Research 1993;71(3):325-333.

[24] Solomon M. Algorithms for the vehicle routing and scheduling problem with time window constraints. Operations Research 1987;35(2):254-265.

[25] Tirado G, Hvattum LM, Fagerholt K, Cordeau J-F. Heuristics for dynamics and stochastic routing in industrial shipping. Computers and Operations Research 2013;40(1):253-263.

Appendix A – TS computational results

Tables A.1 and A.2 show the solution quality comparison among TS variants without a candidate list and Tables A.4 and A.5 present the comparison among TS variants with a candidate list (section 4.4).

Table A.1. Solution quality of TS without candidate list – SIMPLE.Cv and SIMPLE.CvSv

		Best Solution found by TS			% of TS best profit					
					poolMax =5			*poolMax* =10		
		Methods	**Profit**	**Δ % best MIPx**	**TSNone5**	**TSPMF5**	**TSPMD5**	**TSNone10**	**TSPMF10**	**TSPMD10**
Simple.Cv	a	TSNone5,TSPMF5,TSPMD5,TSNone10,TSPMF10,TSPMD10	3343742	0.0%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**
	b	TSNone5,TSPMF5,TSPMD5,TSNone10,TSPMF10,TSPMD10	5410819	0.0%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**
	c	TSNone5,TSPMF5,TSPMD5,TSNone10,TSPMF10,TSPMD10	40605550	0.0%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**
	d	TSNone10,TSPMF10,TSPMD10	42236708	0.0%	0.0%	0.0%	0.0%	**0.0%**	**0.0%**	**0.0%**
	e	TSPMD5	18653004	0.0%	0.0%	0.0%	**0.0%**	-0.7%	-0.7%	0.0%
	f	TSPMD10	77597740	-0.1%	-0.7%	-0.7%	-0.1%	-0.7%	-0.1%	**0.0%**
	g	TSPMD10	44883988	-0.1%	-2.0%	-0.3%	-1.0%	-1.0%	-0.2%	**0.0%**
	h	TSPMD5	62898464	-1.0%	0.0%	0.0%	**0.0%**	0.0%	0.0%	0.0%
	i	TSPMD10	62549573	-0.1%	-0.2%	-0.2%	-0.1%	0.0%	0.0%	**0.0%**
	j	TSPMF5	114999045	-3.5%	-2.5%	**0.0%**	-0.1%	-0.1%	-0.1%	0.0%
	k	TSPMD10	112859379	-0.3%	-0.1%	-0.1%	-0.1%	-0.1%	-0.1%	**0.0%**
	l	TSPMF10	123339799	-0.7%	-0.5%	0.0%	-0.5%	-0.3%	**0.0%**	-0.5%
	m	TSPMF10	132682800	-3.6%	-1.2%	-1.1%	-1.0%	-1.1%	**0.0%**	-0.1%
	n	TSNone5,TSPMF5,TSPMD5,TSNone10,TSPMF10,TSPMD10	172773817	-3.7%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**
Simple.CvSv	a	TSPMF5,TSPMD5,TSNone10,TSPMF10,TSPMD10	5826461	0.0%	0.0%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**
	b	TSPMD5,TSNone10,TSPMF10,TSPMD10	7796802	0.0%	-0.3%	-0.3%	**0.0%**	**0.0%**	**0.0%**	**0.0%**
	c	TSPMD10	47171237	0.0%	-0.6%	-0.6%	-0.6%	-0.6%	-0.6%	**0.0%**
	d	TSPMD5	47558677	-0.3%	-0.1%	-0.1%	**0.0%**	-0.1%	-0.1%	-0.1%
	e	TSPMD10	25898928	0.0%	-3.3%	-3.1%	-1.1%	-2.8%	-2.1%	**0.0%**
	f	TSPMD10	83464896	-1.1%	-0.5%	-0.3%	-0.2%	-0.2%	-0.2%	**0.0%**
	g	TSPMF5	51915539	1.0%	0.0%	**0.0%**	-4.1%	-4.0%	-3.7%	-3.8%
	h	TSPMF10	67595230	-0.9%	-0.3%	-0.3%	-0.9%	-0.3%	**0.0%**	-0.7%
	i	TSPMF10	68507222	-0.4%	-0.4%	-0.2%	0.0%	-0.1%	**0.0%**	-0.1%
	j	TSPMD5	125279120	-1.1%	-2.1%	-2.1%	**0.0%**	-2.2%	-2.2%	-0.6%
	k	TSPMD5	118302835	-0.8%	-0.3%	-0.2%	**0.0%**	-0.1%	0.0%	-0.2%
	l	TSNone10,TSPMF10	131944443	-0.5%	-0.2%	-0.1%	-0.1%	**0.0%**	**0.0%**	-0.1%
	m	TSPMD10	144854899	-1.8%	-1.9%	-1.8%	-1.1%	-1.8%	-1.7%	**0.0%**
	n	TSNone10	184196388	-0.2%	-0.1%	-0.1%	-0.3%	**0.0%**	0.0%	-0.1%
			Minimum	-3.7%	-3.3%	-3.1%	-4.1%	-4.0%	-3.7%	-3.8%
			Average	-0.7%	-0.6%	-0.4%	-0.4%	-0.6%	-0.4%	-0.2%
			Maximum	1.0%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**
			# of best solutions		4	7	11	9	12	15

The first three columns of these tables contain, correspondingly, the name of the variants that found the best feasible solution, the profit of the best solution and the % difference of this profit with the best profit found by any MIP model solution approach of Branchini and Armentano [3]. The cells of these three columns are highlighted with a gray background whenever the profit of the solution found by TS is greater than the profit of the solution found by the MIP model solution approach (e.g. test problem SIMPLE.CvSv *g* of Table A.1). The following 6 columns present the profit percentage difference of the profit found by each TS method and the profit of the best solution found by all 6 TS variants. The last 4 rows present, respectively, the minimum, average and maximum percentage, and the number of times the TS method found a solution equal to or better than the best solution of all 6 variants.

For TS variants without candidate list, TSPMD10 usually produced the best results for SIMPLE.Cv and SIMPLE.CvSv (15 out of 28 test problems of Table A.1, about 54%). In addition, TSPMF5 improved the best known solution of test problem SIMPLE.CvSv *g* by 1.0%. TSPMF10 and TSPMD5 were the variants that produced, respectively, the second and third best results when considering the total number of best solutions found by each method.

For the FULL.Cv and FULL.CvSv test problems (Table A.2), TSPMD10, TSPMD5 and TSPMF10 variants produced the best results. Also, 9 out of the 14 best known solutions of the FULL.CvSv test problems were improved, including the solution of test problem *n*, to which the MIP models approach was not able to find a feasible solution.

As shown in Table A.2, TSPMD10 generated an average result of -0.2%, which is slightly greater than the result of -0.3% of TSPMF10. The profit difference between these two variants is more significant for larger problems, such as FULL test problems *j* to *n*. This indicates that pool management based on distance yields, on average, solutions with marginally higher profits.

Table A.2. Solution quality of TS without candidates list – FULL.Cv and FULL.CvSv

		Best Solution			% of best profit					
					poolMax =5			*poolMax* =10		
		Methods	Profit	Δ % best MIPx	TSNone5	TSPMF5	TSPMD5	TSNone10	TSPMF10	TSPMD10
Full.Cv	a	TSNone5,TSPMF5,TSPMD5,TSNone10,TSPMF10,TSPMD10	51221275	-0.7%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**
	b	TSPMF5,TSPMF10	106088307	-0.9%	-0.5%	**0.0%**	-0.1%	-0.5%	**0.0%**	-0.1%
	c	TSPMD5	108775879	-0.6%	-1.1%	0.0%	**0.0%**	-0.1%	-0.1%	-0.4%
	d	TSNone5,TSPMF5,TSPMD5,TSNone10,TSPMF10	123653652	-1.6%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	-0.1%
	e	TSPMD10	103789092	-0.1%	-0.2%	-0.2%	-0.2%	-0.1%	-0.1%	**0.0%**
	f	TSNone5,TSPMF5,TSPMD5	164734090	-0.4%	**0.0%**	**0.0%**	**0.0%**	-0.1%	-0.1%	-0.1%
	g	TSPMD5	156880611	-2.2%	-0.9%	-0.9%	**0.0%**	-0.9%	-0.9%	-0.9%
	h	TSNone10,TSPMF10	139261461	-1.5%	-0.1%	-0.1%	-0.1%	**0.0%**	**0.0%**	-0.1%
	i	TSPMD5	200550529	-1.2%	-0.3%	-0.2%	**0.0%**	-0.2%	-0.2%	-0.2%
	j	TSPMD10	253292488	-1.2%	-1.1%	-1.1%	-0.6%	-1.1%	-1.1%	**0.0%**
	k	TSNone10,TSPMF10	235517853	-1.6%	-0.2%	-0.2%	-0.1%	**0.0%**	**0.0%**	-0.1%
	l	TSPMD10	246213385	-1.0%	-0.8%	-0.8%	-0.8%	-0.8%	-0.8%	**0.0%**
	m	TSPMD10	317861144	-0.6%	-0.9%	-0.9%	-0.6%	-0.9%	-0.9%	**0.0%**
	n	TSPMD5,TSPMD10	301293459	-1.1%	-0.8%	-0.8%	**0.0%**	-0.1%	-0.1%	**0.0%**
Full.CvSv	a	TSNone5,TSPMF5,TSPMD5,TSNone10,TSPMF10,TSPMD10	53402178	-0.7%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**
	b	TSPMF10	108857004	-1.1%	-0.5%	-0.2%	-0.5%	-0.5%	**0.0%**	-0.5%
	c	TSNone10,TSPMF10	112587365	-1.4%	-0.1%	-0.1%	0.0%	**0.0%**	**0.0%**	-0.1%
	d	TSPMD10	127920763	-1.8%	-0.2%	-0.2%	0.0%	-0.2%	-0.2%	**0.0%**
	e	TSPMD5	106158453	0.7%	-0.5%	-0.5%	**0.0%**	-0.1%	-0.1%	-0.2%
	f	TSPMD5,TSNone10,TSPMF10	170224077	2.5%	-0.5%	-0.5%	**0.0%**	**0.0%**	**0.0%**	-0.2%
	g	TSNone5,TSPMF5,TSPMD5	163035362	-0.3%	**0.0%**	**0.0%**	**0.0%**	-2.1%	-2.1%	-2.1%
	h	TSNone5,TSPMF5,TSNone10,TSPMF10	143935991	0.7%	**0.0%**	**0.0%**	-0.1%	**0.0%**	**0.0%**	-0.2%
	i	TSPMD10	205689641	0.7%	-0.1%	-0.1%	-0.1%	-0.1%	-0.1%	**0.0%**
	j	TSPMF10	259737301	27.9%	-0.6%	-0.6%	-0.9%	-0.4%	**0.0%**	-0.2%
	k	TSPMD10	238066869	6.5%	-0.1%	-0.1%	-0.3%	-0.1%	-0.1%	**0.0%**
	l	TSPMD10	253667953	27.2%	-0.7%	-0.6%	-1.1%	-1.1%	-1.1%	**0.0%**
	m	TSPMD10	326642027	17.9%	-0.4%	-0.4%	-0.2%	-0.4%	-0.4%	**0.0%**
	n	TSPMD10	309827504	na	-0.1%	-0.1%	-0.4%	-0.1%	-0.1%	**0.0%**
				Minimum						
			Minimum	-2.2%	-1.1%	-1.1%	-1.1%	-2.1%	-2.1%	-2.1%
			Average	2.4%	-0.4%	-0.3%	-0.2%	-0.4%	-0.3%	-0.2%
			Maximum	27.9%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**
			# of best solutions		6	7	11	8	11	13

na: not available because no MIPx method found a feasible solution for test problem FULL.CvSv n

Additionally, because of the pool management based on distance, pool solutions of PMD variants are more diverse than those of PMF variants. For example, both TSPMF10 and TSPMD10 variants found the same best solution for test problem FULL.Cv *i*. However, as shown in Figure A.1, the pool of TSPMD10 is more diverse than the pool of TSPMF10 for this test problem. Figure A.1 presents, for each solution out of the 10 solutions of the pool of each method, the profit percentage difference of a solution and the best solution (y-axis) and the minimum distance between the solution and all other solutions of the pool (x-axis).

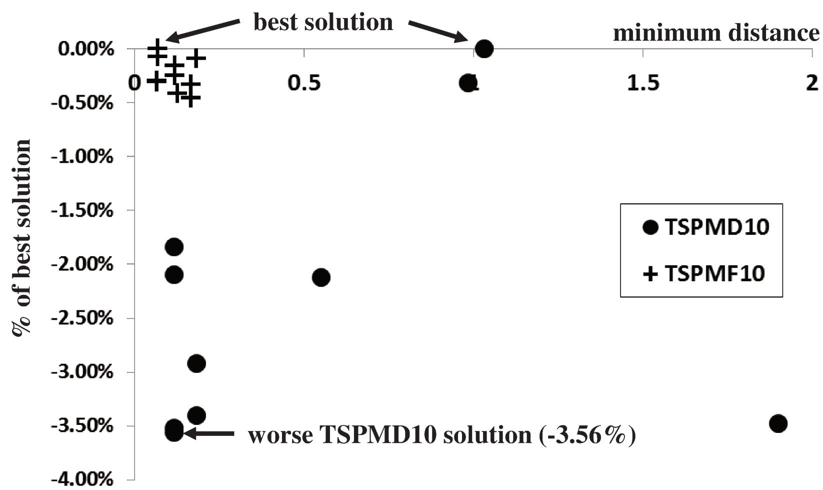


Figure A.1. Solution quality and distance of TSPMD10 and TSPMF10 - test problem FULL.Cv *i*

Figure A.1 shows that, except for the two best solutions of TSPMD10, all other 8 solutions of the TSPMD10 pool have worse quality than those of TSPMF10 pool (the worst profit is -3.56% of the best profit). On the other hand, the minimum distance[6] of TSPMD10 (around 0.12) is greater than the minimum distance of TSPMF10 (around 0.07) and, also, TSPMD10 pool contains 4 solutions with minimum distance above 0.5. These greater distances show that the diversity of the TSPMD10 pool is much greater than that of TSPMF10. Even though both variants found the same best solution for the test problem, this example clearly shows the different goals of each pool management method. The pool of TSPMF10 has, on average, greater solution quality and lower minimum distance. On the other hand, the pool of TSPMD10 has, on average, lower solution quality and higher minimum distance.

Another conclusion drawn from Tables A.1 and A.2 is that, in general, the variants with *poolMax*=10 generated results slightly better than those of the variants with *poolMax*=5. Therefore, one must balance if the better quality is worth the additional computational time of *poolMax*=10 variants. The additional computational time of *poolMax*=10 variants over *poolMax*=5 variants is shown in Table A.3.

The first column of Table A.3 shows the computational time of TSNone5 for the SIMPLE.Cv and SIMPLE.CvSv test problems. The next three columns present the computational time *ratio* between variants with *poolMax*=10 and variants with *poolMax*=5

[6] Note that if two solutions points of Figure A.1 are close to each other, it does not mean that the respective two solutions are similar. Two points close to each other just means that both solutions have similar profit and similar minimum distance measure among all other solutions of the pool.

$\left(ratio = \frac{Computational\ Time\ of\ poolMax=10}{Computational\ Time\ of\ poolMax=5}\right)$. The last four columns show the same information for the FULL.Cv and FULL.CvSv test problems.

Table A.3. Ratio of the computational time between *poolMax*=10 and *poolMax*=5 variants

	SIMPLE	TSNone5 (sec)	Ratio TSNone	Ratio TSPMF	Ratio TSPMD	FULL	TSNone5 (sec)	Ratio TSNone	Ratio TSPMF	Ratio TSPMD
Cv	a	100	1.5	1.7	1.9	a	156	1.7	1.8	2.0
Cv	b	145	1.6	1.7	1.6	b	291	1.4	1.5	3.6
Cv	c	249	1.9	2.1	1.9	c	554	2.7	1.9	1.4
Cv	d	379	1.5	1.7	1.8	d	517	1.6	1.6	2.0
Cv	e	370	1.7	1.6	1.9	e	769	1.8	2.3	1.8
Cv	f	582	1.3	1.4	1.7	f	874	1.9	1.9	2.4
Cv	g	821	1.3	1.8	1.9	g	1,396	1.4	1.6	1.3
Cv	h	868	1.4	1.2	2.0	h	1,777	1.4	1.7	2.3
Cv	i	916	1.4	1.6	2.1	i	2,173	1.6	1.8	2.2
Cv	j	1,555	1.6	1.5	1.8	j	4,617	1.7	1.8	2.7
Cv	k	1,764	1.4	1.5	1.9	k	4,439	2.2	2.4	1.5
Cv	l	2,193	1.5	1.6	2.2	l	5,318	1.9	2.0	2.6
Cv	m	2,846	1.3	1.4	2.0	m	9,238	1.9	1.8	2.1
Cv	n	2,918	1.6	1.7	1.8	n	9,305	2.1	1.9	2.3
CvSv	a	135	1.9	1.6	1.6	a	214	1.3	1.5	1.9
CvSv	b	228	2.0	1.9	1.9	b	229	1.6	1.6	1.5
CvSv	c	455	1.5	1.8	2.3	c	532	2.1	2.0	2.0
CvSv	d	529	3.0	1.8	1.9	d	446	1.8	1.9	2.3
CvSv	e	984	1.3	1.6	1.9	e	608	2.4	2.0	2.0
CvSv	f	1,150	2.2	1.7	1.6	f	903	1.7	1.8	2.6
CvSv	g	1,822	1.2	1.7	1.7	g	1,070	1.3	1.6	1.3
CvSv	h	2,010	1.1	1.8	1.8	h	1,399	2.1	2.1	2.6
CvSv	i	1,054	2.1	1.9	1.6	i	1,691	1.5	2.0	1.7
CvSv	j	4,464	1.0	1.8	2.0	j	4,296	1.9	2.3	2.1
CvSv	k	3,299	1.6	1.8	1.8	k	3,652	1.8	1.8	2.7
CvSv	l	3,161	1.6	2.0	2.2	l	6,489	1.7	1.5	2.1
CvSv	m	5,415	2.3	1.8	1.9	m	9,300	1.4	1.5	1.4
CvSv	n	7,384	2.0	2.1	1.9	n	9,259	1.9	2.0	1.6
	Minimum	100	1.0	1.2	1.6	Minimum	156	1.3	1.5	1.3
	Average	1,707	1.6	1.7	1.9	Average	2,911	1.8	1.8	2.1
	Maximum	7,384	3.0	2.1	2.3	Maximum	9,305	2.7	2.4	3.6

Table A.3 shows that TSNone5 has a computational time between 100 seconds and 2.6 hours and that the computational time of *poolMax*=10 variants is, on average, around twice the computational time of *poolMax*=5 variants.

Therefore, the combined analysis of Tables A.1, A.2 and A.3 shows that the results of method TSPMF5 of Table A.1 and the results of variants TSPMF5 and TSPMD5 of Table A.2 are very competitive when compared with those of TSPMD10. TSPMD10 requires about the double of the computational time of these variants with *poolMax*=5 to achieve marginal improvements in the quality of the solutions.

The results with candidate list, presented in Tables A.4 and A.5, reveal that TSlcPMD10 produced the best results among all variants. TSlcPMD10 generated the best results in 18 out of the 28 SIMPLE.Cv and SIMPLE.CvSv (64%) and in 17 out of the 28 FULL.Cv and FULL.CvSv test problems (61%). Table A.5 also shows that TS variants with candidate list improved the best known results of the same 9 FULL.CvSv test problems of Table A.2.

Table A.4. Solution quality of TS with candidate list – SIMPLE.Cv and SIMPLE.CvSv

		Best Solution found by TS			% of TS best profit					
				Δ % best	*poolMax* =5			*poolMax* =10		
		Methods	Profit	MIPx	TSlcNone5	TSlcPMF5	TSlcPMD5	TSlcNone10	TSlcPMF10	TSlcPMD10
Simple.Cv	a	TSlcNone5,TSlcPMF5,TSlcPMD5,TSlcNone10,TSlcPMF10,TSlcPMD10	3343742	0.0%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**
	b	TSlcNone5,TSlcPMF5,TSlcPMD5,TSlcNone10,TSlcPMF10,TSlcPMD10	5410819	0.0%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**
	c	TSlcNone5,TSlcPMF5,TSlcPMD5,TSlcNone10,TSlcPMF10,TSlcPMD10	40605550	0.0%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**
	d	TSlcPMD10	42217121	0.0%	-0.6%	-0.1%	-0.2%	-0.6%	-0.1%	**0.0%**
	e	TSlcPMD10	18648579	0.0%	-0.6%	-0.6%	-0.8%	-0.6%	-0.5%	**0.0%**
	f	TSlcNone10,TSlcPMF10	77686919	0.0%	-0.8%	-0.8%	-0.2%	**0.0%**	**0.0%**	-0.2%
	g	TSlcPMD10	44916802	0.0%	-3.6%	-0.1%	0.0%	-3.6%	-3.6%	**0.0%**
	h	TSlcPMD10	62898464	-1.0%	0.0%	0.0%	0.0%	0.0%	0.0%	**0.0%**
	i	TSlcPMD5	62481329	-0.2%	-0.2%	-0.1%	**0.0%**	0.0%	0.0%	-0.2%
	j	TSlcPMD10	114953417	-3.5%	-1.5%	-1.5%	0.0%	-1.5%	-1.5%	**0.0%**
	k	TSlcPMD10	113017701	-0.2%	-0.3%	-0.3%	-0.3%	0.0%	0.0%	**0.0%**
	l	TSlcPMD10	123263787	-0.7%	-0.3%	-0.3%	-0.3%	-0.3%	-0.3%	**0.0%**
	m	TSlcPMD10	133446774	-3.1%	-0.9%	-0.9%	-0.3%	-1.0%	-0.9%	**0.0%**
	n	TSlcNone5,TSlcPMF5	172713985	-3.7%	**0.0%**	**0.0%**	-0.3%	-0.3%	-0.1%	0.0%
Simple.CvSv	a	TSlcPMD5,TSlcPMD10	5826461	0.0%	0.0%	0.0%	**0.0%**	0.0%	0.0%	**0.0%**
	b	TSlcPMD10	7796802	0.0%	-0.3%	-0.3%	-0.3%	-0.3%	-0.3%	**0.0%**
	c	TSlcPMF5,TSlcNone10,TSlcPMF10,TSlcPMD10	46901316	-0.6%	0.0%	**0.0%**	0.0%	**0.0%**	**0.0%**	**0.0%**
	d	TSlcNone10,TSlcPMF10,TSlcPMD10	46604329	-2.3%	-0.3%	-0.2%	-0.2%	**0.0%**	**0.0%**	**0.0%**
	e	TSlcPMD10	25702817	-0.8%	-1.6%	-0.4%	-1.9%	-1.8%	-1.6%	**0.0%**
	f	TSlcPMD10	83388381	-1.2%	-0.6%	-0.6%	-1.4%	-0.7%	-0.5%	**0.0%**
	g	TSlcNone5,TSlcPMF5	50022639	-2.7%	**0.0%**	**0.0%**	-0.4%	-0.6%	-0.6%	-0.6%
	h	TSlcNone5,TSlcPMD10	68003256	-0.3%	**0.0%**	-0.2%	-0.9%	-0.4%	-0.4%	**0.0%**
	i	TSlcNone5,TSlcPMF5,TSlcPMD5,TSlcNone10,TSlcPMF10	68402550	-0.6%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	0.0%
	j	TSlcNone10,TSlcPMF10	122834707	-3.0%	-0.3%	-0.2%	-0.6%	**0.0%**	**0.0%**	-0.4%
	k	TSlcPMD5	118350049	-0.7%	-0.2%	-0.1%	**0.0%**	-0.3%	-0.3%	-0.1%
	l	TSlcPMF10	132244571	-0.3%	-0.2%	-0.2%	-0.2%	-0.1%	**0.0%**	-0.2%
	m	TSlcPMD5	144248865	-2.2%	-0.3%	-0.3%	**0.0%**	-0.4%	-0.4%	-0.4%
	n	TSlcPMF5	184291330	-0.1%	-0.1%	**0.0%**	-0.4%	-0.2%	-0.2%	-0.2%
			Minimum	-3.7%	-3.6%	-1.5%	-1.9%	-3.6%	-3.6%	-0.6%
			Average	-1.0%	-0.5%	-0.3%	-0.3%	-0.5%	-0.4%	-0.1%
			Maximum	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**
				# of best solutions	7	8	8	8	9	18

However, when considering the additional computational time of variants with *poolMax*=10 over variants with *poolMax*=5, TSlcPMF5 is the best alternative to the SIMPLE problem models (average result of -0.3%, minimum of -1.5%) and TSlcPMD5 is the best option for the FULL problem models (average result of -0.2%, minimum of -1.0%).

Table A.5. Solution quality of TS with candidate list – FULL.Cv and FULL.CvSv

		Best Solution			% of best profit					
				Δ % best	*poolMax* =5			*poolMax* =10		
		Methods	Profit	MIPx	TSlcNone5	TSlcPMF5	TSlcPMD5	TSlcNone10	TSlcPMF10	TSlcPMD10
Full.Cv	a	TSlcNone5,TSlcPMF5,TSlcPMD5,TSlcNone10,TSlcPMF10,TSlcPMD10	51221275	-0.7%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**
	b	TSlcPMD5,TSlcNone10,TSlcPMF10	106088307	-0.9%	-0.4%	-0.1%	**0.0%**	**0.0%**	**0.0%**	-0.1%
	c	TSlcPMF10	108691411	-0.6%	-1.0%	-1.0%	-0.4%	0.0%	**0.0%**	-0.2%
	d	TSlcNone10,TSlcPMF10,TSlcPMD10	123679184	-1.6%	0.0%	0.0%	-0.4%	**0.0%**	**0.0%**	**0.0%**
	e	TSlcPMD5	103741420	-0.1%	-0.4%	-0.4%	**0.0%**	-0.1%	-0.1%	0.0%
	f	TSlcPMD10	164678230	-0.4%	0.0%	0.0%	-0.2%	0.0%	0.0%	**0.0%**
	g	TSlcPMD5,TSlcPMD10	157052409	-2.1%	-0.1%	-0.1%	**0.0%**	-0.1%	-0.1%	**0.0%**
	h	TSlcPMD10	140609739	-0.5%	-0.6%	-0.6%	-0.6%	-0.6%	-0.6%	**0.0%**
	i	TSlcNone5,TSlcPMF5,TSlcPMD5,TSlcN	199835329	-1.5%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**
	j	TSlcNone10,TSlcPMF10	253829401	-1.0%	-0.7%	-0.7%	-0.8%	**0.0%**	**0.0%**	-0.2%
	k	TSlcPMD5	235337241	-1.7%	-0.6%	-0.6%	**0.0%**	-0.6%	-0.6%	-0.1%
	l	TSlcPMD10	245407642	-1.3%	-1.0%	-1.0%	-1.0%	-1.0%	-1.0%	**0.0%**
	m	TSlcPMD5	316908357	-0.9%	-0.6%	-0.6%	**0.0%**	-0.1%	-0.1%	-0.1%
	n	TSlcPMD10	301246527	-1.1%	-0.3%	-0.3%	-0.3%	-0.3%	-0.3%	**0.0%**
Full.CvSv	a	TSlcNone5,TSlcPMF5,TSlcPMD5,TSlcNone10,TSlcPMF10,TSlcPMD10	53402178	-0.7%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**
	b	TSlcPMF10	108287065	-1.6%	-0.6%	-0.1%	0.0%	-0.5%	**0.0%**	-0.2%
	c	TSlcNone10,TSlcPMF10	112631656	-1.4%	0.0%	0.0%	0.0%	**0.0%**	**0.0%**	0.0%
	d	TSlcPMD10	128296392	-1.6%	-0.6%	-0.6%	-0.4%	0.0%	0.0%	**0.0%**
	e	TSlcNone5,TSlcPMF5,TSlcPMD5,TSlcNone10,TSlcPMF10,TSlcPMD10	106294800	0.8%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**
	f	TSlcNone5,TSlcPMF5,TSlcPMD5,TSlcNone10,TSlcPMF10,TSlcPMD10	170337072	2.6%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**
	g	TSlcPMD10	162895067	-0.4%	-1.7%	-1.7%	-0.4%	-1.7%	-1.7%	**0.0%**
	h	TSlcPMD10	144343250	1.0%	-0.6%	-0.6%	-0.5%	-0.6%	-0.6%	**0.0%**
	i	TSlcNone5,TSlcPMF5,TSlcNone10,TSlcPMF10,TSlcPMD10	205491512	0.6%	**0.0%**	**0.0%**	-0.1%	**0.0%**	**0.0%**	**0.0%**
	j	TSlcPMD5	258821202	27.4%	-0.3%	-0.3%	**0.0%**	-0.3%	-0.3%	-0.1%
	k	TSlcPMF10	238135783	6.5%	-0.4%	-0.4%	-0.2%	-0.1%	**0.0%**	-0.3%
	l	TSlcPMD5	253208233	27.0%	-0.9%	-0.9%	**0.0%**	-1.4%	-0.9%	-0.6%
	m	TSlcPMD10	324897560	17.2%	-0.1%	-0.1%	0.0%	0.0%	0.0%	**0.0%**
	n	TSlcPMD10	310174265	na	-0.3%	-0.3%	-0.2%	-0.2%	-0.2%	**0.0%**
			Minimum	-2.1%	-1.7%	-1.7%	-1.0%	-1.7%	-1.7%	-0.6%
			Average	2.3%	-0.4%	-0.4%	-0.2%	-0.3%	-0.2%	-0.1%
			Maximum	27.4%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.0%**
			# of best solutions		6	6	12	10	13	17

na: not available because no MIPx method found a feasible solution for test problem FULL.CvSv n

Next, Figure A.2 compares the percentage of the best solutions, which is the number of best solutions found by the method divided by the total number of test problems (56 test problems), of variants with *poolMax*=5 and *poolMax*=10.

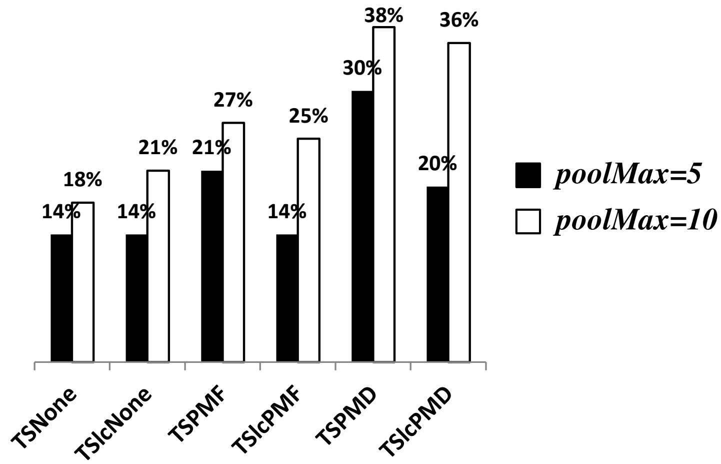


Figure A.2. Percentage of best solutions of variants with *poolMax=5* and *poolMax=10*

Figure A.2 shows that the percentage of best solutions of variants with *poolMax*=10 is always higher than that of variants with *poolMax*=5. The figure also reveals that the TSPMD10 and TSlcPMD10 variants were the best performers with a percentage of the best solutions of 38% and 36%, respectively.

Another conclusion drawn from Figure A.2 is that, except for TSNone10, the use of a candidate list did not enhance the overall performance of TS. Table A.6 presents the detailed results of the best TS variants, TSPMD10 (without candidate list) and TSlcPMD10 (with candidate list), and highlights when the use of a candidate list demonstrated to be advantageous.

The first two columns of Table A.6 show, for the SIMPLE.Cv and SIMPLE.CvSv test problems, the profit of TSPMD10 and TSlcPMD10, respectively. The third column is the percentage difference between the profit of TSlcPMD10 and TSPMD10. When the profit of TSlcPMD10 is greater than the profit of TSPMD10, the percentage is greater than zero, and the table cell is highlighted with a gray background. The fourth column is the ratio of the computational time, which is equal to $\frac{\textit{computational time of TSlcPMD10}}{\textit{computational time of TSPMD10}}$. The last four columns present the same information for the FULL.Cv and FULL.CvSv test problems.

Table A.6 shows that, out of the 56 test problems, the use of a candidate list improved the results of 23 test problems, reached the same results in 7 test problems, and performed worse in 26 problems. In addition, the average difference is between -0.1% and 0.1%, which suggests that it is not possible to find a significant difference between the quality of the solutions found by each method. Moreover, since both TS variants use the same stop criterion, which is based on the total number of calls of the procedure that calculates *route_profit*$_r$, as described in section 4.1, the average computational times of both variants are likely to be close, as shown by the columns ‘Comp. time ratio’.

Table A.6. Comparison of the results of TSPMD10 and TSlcPMD10

		Profit			Comp.		Profit			Comp.
	SIMPLE	TSPMD10	TSlcPMD10	Δ (%)	time ratio	FULL	TSPMD10	TSlcPMD10	Δ (%)	time ratio
Cv	a	3,343,742	3,343,742	0.000%	1.0	a	51,221,275	51,221,275	0.000%	0.8
	b	5,410,819	5,410,819	0.000%	1.2	b	105,980,205	106,007,202	**0.025%**	0.7
	c	40,605,550	40,605,550	0.000%	1.0	c	108,311,888	108,450,882	**0.128%**	1.2
	d	42,236,708	42,217,121	-0.046%	0.8	d	123,588,565	123,679,184	**0.073%**	0.8
	e	18,645,007	18,648,579	**0.019%**	0.9	e	103,789,092	103,725,208	-0.062%	1.3
	f	77,597,740	77,497,131	-0.130%	1.1	f	164,623,229	164,678,230	**0.033%**	0.7
	g	44,883,988	44,916,802	**0.073%**	1.2	g	155,396,780	157,052,409	**1.065%**	1.3
	h	62,897,844	62,898,464	**0.001%**	1.0	h	139,162,956	140,609,739	**1.040%**	1.0
	i	62,549,573	62,376,115	-0.277%	1.0	i	200,110,023	199,835,329	-0.137%	0.9
	j	114,998,250	114,953,417	-0.039%	1.0	j	253,292,488	253,387,454	**0.037%**	0.7
	k	112,859,379	113,017,701	**0.140%**	1.3	k	235,191,798	235,048,211	-0.061%	0.9
	l	122,746,937	123,263,787	**0.421%**	0.9	l	246,213,385	245,407,642	-0.327%	1.1
	m	132,498,081	133,446,774	**0.716%**	1.2	m	317,861,144	316,486,209	-0.433%	0.9
	n	172,773,817	172,644,406	-0.075%	1.4	n	301,293,459	301,246,527	-0.016%	0.9
CvSv	a	5,826,461	5,826,461	0.000%	1.2	a	53,402,178	53,402,178	0.000%	1.0
	b	7,796,802	7,796,802	0.000%	1.0	b	108,312,075	108,098,179	-0.197%	1.3
	c	47,171,237	46,901,316	-0.572%	0.9	c	112,450,273	112,589,445	**0.124%**	1.2
	d	47,514,281	46,604,329	-1.915%	1.0	d	127,920,763	128,296,392	**0.294%**	0.9
	e	25,898,928	25,702,817	-0.757%	1.0	e	105,986,419	106,294,800	**0.291%**	1.0
	f	83,464,896	83,388,381	-0.092%	1.0	f	169,889,570	170,337,072	**0.263%**	1.0
	g	49,928,732	49,704,871	-0.448%	1.1	g	159,689,986	162,895,067	**2.007%**	1.7
	h	67,129,373	68,003,256	**1.302%**	1.1	h	143,626,732	144,343,250	**0.499%**	0.8
	i	68,445,444	68,400,202	-0.066%	1.0	i	205,689,641	205,491,512	-0.096%	1.0
	j	124,551,066	122,290,939	-1.815%	0.9	j	259,160,648	258,553,001	-0.234%	0.9
	k	118,075,399	118,237,740	**0.137%**	1.0	k	238,066,869	237,341,285	-0.305%	0.8
	l	131,758,581	132,015,086	**0.195%**	1.0	l	253,667,953	251,793,075	-0.739%	0.9
	m	144,854,899	143,662,116	-0.823%	0.8	m	326,642,027	324,897,560	-0.534%	1.4
	n	184,041,563	183,946,208	-0.052%	1.1	n	309,827,504	310,174,265	**0.112%**	0.8
			Minimum	-1.915%	0.8			**Minimum**	-0.739%	0.7
			Average	-0.147%	1.0			**Average**	0.102%	1.0
			Maximum	1.302%	1.4			**Maximum**	2.007%	1.7

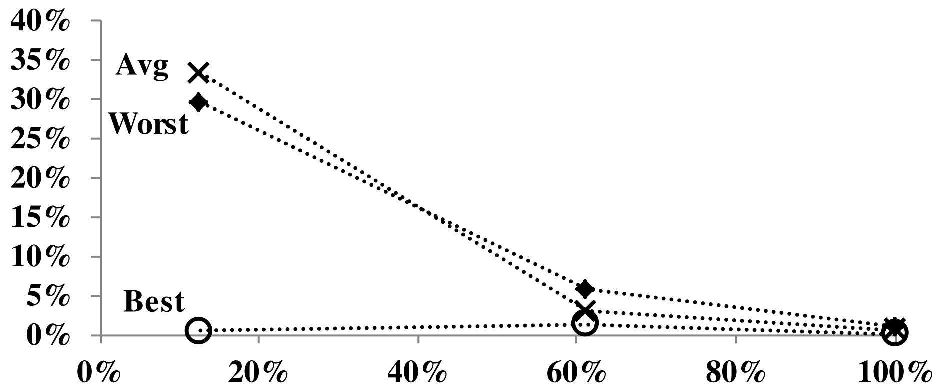
Table A.6 also shows that the profits of all test problems with spot voyages (.CvSv) are larger than the profits without spot voyages (.Cv). On average, the profits of the SIMPLE.CvSv test problems are 18% larger than that of SIMPLE.Cv and the profits of the FULL.CvSv test problems are 3% larger than that of FULL.Cv. The profits of the FULL test problems are also always larger than the profits of the respective SIMPLE test problems (at least 68% larger). These results are explained by the use of cheaper hired ships instead of owned ships, and the capture of additional charter revenues, as detailed in the last columns of Table A.7 for the FULL.CvSv test problems.

Table A.7. Fleet composition of the best results of the FULL.CvSv problems

		Problem size			Best Feasible Solution		Number of own ships				Number of hired in ships		Chartered out revenue
							Used to serve voyages		Not used to serve voyages				
		Ships	Cvs	Svs	Methods	Profit	No charter out	With charter out	Charter out	Lay-up	Used	Not used	
Full.CvSv	a	24	30	10	MIPE,MIPBN15	53797761	5	4	2	7	0	6	76947721
	b	25	30	10	MIPE,MIPBN15	110047995	3	6	4	8	1	3	146317712
	c	34	35	13	MIPE	114180627	3	7	0	18	0	6	110871249
	d	42	40	13	MIPBN15	130331112	3	12	1	14	1	11	150218828
	e	41	45	16	TSlcNone5-10, TSlcPMF5-10, TSlcPMD5-10	106294800	4	8	0	22	1	6	159827501
	f	56	50	16	TSlcNone5-10, TSlcPMF5-10, TSlcPMD5-10	170337072	7	8	3	18	0	20	189707023
	g	48	60	19	MIPBN00	163554256	5	12	1	18	5	7	205071906
	h	49	60	19	TSlcPMD10	144343250	7	9	0	22	1	10	164890712
	i	60	60	19	TSPMD10	205689641	3	11	2	22	5	17	224543497
	j	70	90	28	TSPMF10	259737301	8	11	4	23	0	24	242567005
	k	70	90	28	TSlcPMF10	238135783	8	15	0	23	1	23	212883723
	l	70	90	28	TSPMD10	253667953	7	15	1	23	2	22	240363177
	m	82	110	34	TSPMD10	326642027	6	17	0	27	3	29	282268534
	n	82	110	34	TSlcPMD10	310174265	8	16	1	25	2	30	244090608

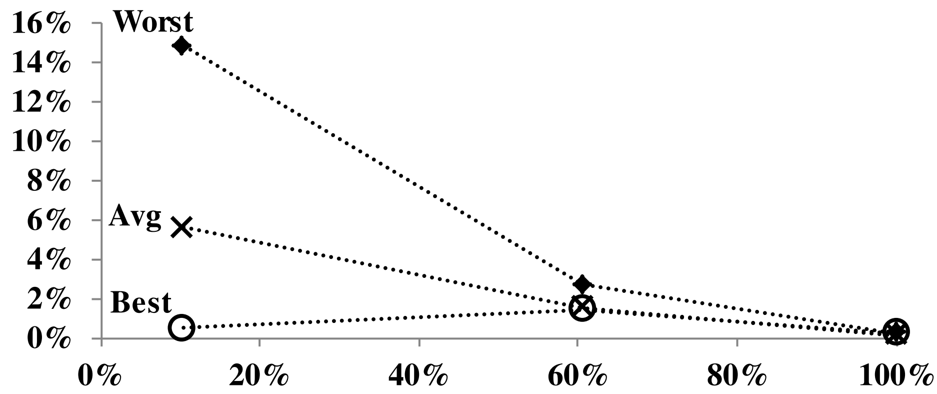
Table A.7 shows the fleet composition of the best results of the FULL.CvSv problems. The sum of each row of the six columns of Table A.7 that present the number of ships per type is equal to the number of ships of the first column. The next two columns, 'Cvs' and 'Svs', present, respectively, the total number of contractual and spot voyages of each test problem. The TS variants showed in the gray shaded cells improved the best known profit of 9 FULL.CvSv problems (*e*, *f* and *h* to *n*).

To investigate the individual contribution to the solution quality improvement of each TS phase of Figure 5, Figures A.3, A.4 and A5 show, respectively, the performance of the TSPMF10, TSPMD10 and TSlcPMD10 variants for the SIMPLE.CvSv and FULL.CvSv test problems. The set of the 28 SIMPLE.CvSv and FULL.CvSv test problems was selected because this is the set in which the results yielded by TS variants are competitive when compared with the results obtained by MIPE considering either the solution quality or the computational time criteria. The y-axis shows the % relative improvement of the solution quality between two subsequent phases and the x-axis shows the percentage of the total time that was required to achieve the result at the end of each phase. The figures show three lines, one for the average quality of pool solutions and another two lines for the worst and best solutions of the pool. For example, the phase 2 of TSPMF10 (Figure A.3) improved, on average, the solution quality of the pool by 33.36%. This result is obtained at about 13% of the total time. Then, phase 3 improved the average quality of the pool solutions found by phase 2 by an additional 2.95%. To achieve this result, phases 1, 2 and 3 required 61% of the total time. Lastly, phase 4 improved the latter result by 0.53%.



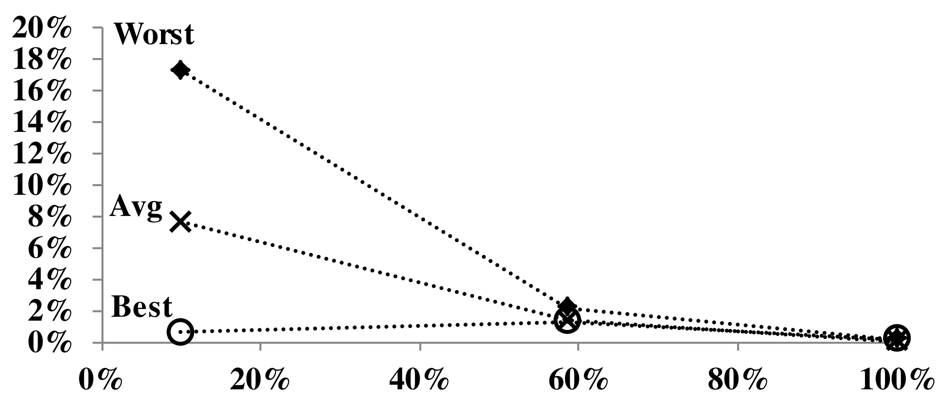
Phases	Time	Best	Avg	Worst
From 1 to 2	13%	0.48%	33.36%	29.61%
From 2 to 3	61%	1.29%	2.95%	5.67%
From 3 to 4	100%	0.12%	0.53%	0.95%

Figure A.3. TSPMF10 performance, per TS phase, for the SIMPLE.CvSv and FULL.CvSv test problems



Phases	Time	Best	Avg	Worst
From 1 to 2	10%	0.47%	5.62%	14.82%
From 2 to 3	61%	1.41%	1.55%	2.71%
From 3 to 4	100%	0.22%	0.14%	0.25%

Figure A.4. TSPMD10 performance, per TS phase, for the SIMPLE.CvSv and FULL.CvSv test problems



Phases	Time	Best	Avg	Worst
From 1 to 2	10%	0.62%	7.64%	17.31%
From 2 to 3	59%	1.27%	1.38%	2.17%
From 3 to 4	100%	0.15%	0.02%	0.15%

Figure A.5. TSlcPMD10 performance, per TS phase, for the SIMPLE.CvSv and FULL.CvSv test problems

These three figures show that phase 2 spends around 10% of the total computational time, and is responsible for a significant improvement in the quality of the worst solution, as well as in the average quality of the pool solutions. In this phase, however, the best solution is only improved around 0.5%. Subsequently, phase 3 is responsible for the greatest improvement of the best solution of the pool. This improvement is between 1.27% and 1.41% and is completed at roughly 60% of the total computational time. Finally, except for the improvement of 0.95% of the worst solution of TSPMF10 shown in Figure A.3, phase 4 marginally improves the quality of pool solutions.

In spite of the fact that the results presented in Figures A.3, A.4 and A.5 are restricted to the SIMPLE.CvSv and FULL.CvSv test problems and to the TSPMF10, TSPMD10 and, TSlcPMD10 variants, similar conclusions are obtained from the analysis of results generated by the other variants when applied to the remaining set of test problems without spot voyages.

Appendix B – Computational results of the constructive heuristic

Although TS does not need a feasible solution to start with, the constructive heuristic of section 4.2 was able to find feasible solutions for all 56 test problems in a computational time of a few seconds, as shown in Table B.1.

Table B.1. Constructive heuristic results

	SIMPLE	Feasible?	Profit	% of best profit	CPU Time (sec.)	FULL	Feasible?	Profit	% of best profit	CPU Time (sec.)
Cv	a	yes	-18509348	-654%	0.02	a	yes	8157268	-84%	0.03
	b	yes	-14456821	-367%	0.02	b	yes	56455606	-47%	0.02
	c	yes	6376424	-84%	0.05	c	yes	65446443	-40%	0.05
	d	yes	-2921990	-107%	0.08	d	yes	80738614	-35%	0.09
	e	yes	-30448861	-263%	0.16	e	yes	34592507	-67%	0.16
	f	yes	40425327	-48%	0.22	f	yes	87999259	-47%	0.23
	g	yes	-19243640	-143%	0.28	g	yes	82981455	-47%	0.30
	h	yes	-10650212	-117%	0.29	h	yes	75200651	-47%	0.32
	i	yes	5747933	-91%	0.27	i	yes	139400061	-30%	0.30
	j	yes	64136634	-44%	1.03	j	yes	166861711	-34%	1.17
	k	yes	51822020	-54%	0.89	k	yes	125748114	-47%	1.04
	l	yes	49143616	-60%	1.00	l	yes	183042690	-26%	1.17
	m	yes	41686988	-69%	1.56	m	yes	160025021	-50%	1.73
	n	yes	108851083	-37%	1.49	n	yes	176167802	-42%	1.66
CvSv	a	yes	-18509348	-418%	0.02	a	yes	8157268	-85%	0.03
	b	yes	-14456821	-285%	0.02	b	yes	56455606	-48%	0.02
	c	yes	6376424	-86%	0.05	c	yes	65446443	-42%	0.06
	d	yes	-2921990	-106%	0.08	d	yes	80738614	-37%	0.09
	e	yes	-30448861	-218%	0.16	e	yes	34592507	-67%	0.16
	f	yes	40425327	-52%	0.22	f	yes	87999259	-48%	0.23
	g	yes	-19243640	-137%	0.28	g	yes	82981455	-49%	0.30
	h	yes	-10650212	-116%	0.29	h	yes	75200651	-48%	0.31
	i	yes	5747933	-92%	0.27	i	yes	139400061	-32%	0.30
	j	yes	64136634	-49%	1.02	j	yes	166861711	-36%	1.18
	k	yes	51822020	-56%	0.88	k	yes	125748114	-47%	1.08
	l	yes	49143616	-63%	1.03	l	yes	183042690	-28%	1.20
	m	yes	41686988	-71%	1.56	m	yes	160025021	-51%	1.75
	n	yes	108851083	-41%	1.48	n	yes	176167802	-43%	1.66
			Minimum	-654%	0.02			**Minimum**	-85%	0.02
			Average	-140%	0.53			**Average**	-47%	0.59
			Maximum	-37%	1.56			**Maximum**	-26%	1.75

The left part of Table B.1 shows the constructive heuristic results for the SIMPLE.Cv and SIMPLE.CvSv test problems and the right part of the table presents the results for the FULL.Cv and FULL.CvSv test problems. The column '% of best profit' is the percentage profit difference, between the profit found by the constructive heuristic and the best profit found by all TS variants.

The constructive heuristic found feasible solutions for all 56 test problems. Solution quality was poor, especially for the SIMPLE.Cv and SIMPLE.CvSv test problems. In these problems, the profit percentage of the solution found by the constructive heuristic could be from -654% to -37% worse than the best solution found by TS variants. Nonetheless, the constructive heuristic is very fast, with the execution time of at most 1.75 seconds.

Despite the poor quality of the solutions generated by the constructive heuristic, it achieved the goal of constructing, in a very short period of time, a starting solution for TS.