Master's degree thesis

LOG950 Logistics

Periodic Supply Vessel Planning with Flexible Departures and Coupled Vessels

Aliaksei Liasnoi

Number of pages including this page: 96

Molde, 2015

Molde University College
Specialized University in Logistics

Mandatory statement

Each student is responsible for complying with rules and regulations that relate to examinations and to academic work in general. The purpose of the mandatory statement is to make students aware of their responsibility and the consequences of cheating. Failure to complete the statement does not excuse students from their responsibility.

Please complete the mandatory statement by placing a mark ***in each box*** *for statements 1-6 below.*		
1.	**I/we hereby declare that my/our paper/assignment is my/our own work, and that I/we have not used other sources or received other help than mentioned in the paper/assignment.**	☒
2.	**I/we hereby declare that this paper** 1. Has not been used in any other exam at another department/university/university college 2. Is not referring to the work of others without acknowledgement 3. Is not referring to my/our previous work without acknowledgement 4. Has acknowledged all sources of literature in the text and in the list of references 5. Is not a copy, duplicate or transcript of other work	Mark each box: 1. ☒ 2. ☒ 3. ☒ 4. ☒ 5. ☒
3.	I am/we are aware **that any breach of the above will be considered as cheating, and may result in annulment of the examination and exclusion from all universities and university colleges in Norway for up to one year, according to the Act relating to Norwegian Universities and University Colleges, section 4-7 and 4-8 and Examination regulations** section 14 and 15.	☒
4.	**I am/we are aware that all papers/assignments may be checked for plagiarism by a software assisted plagiarism check**	☒
5.	**I am/we are aware that Molde University College will handle all cases of suspected cheating according to prevailing guidelines.**	☒
6.	**I/we are aware of the University College's rules and regulation for using sources**	☒

Publication agreement

ECTS credits: 30

Supervisor: Professor Irina Gribkovskaia

Agreement on electronic publication of master thesis

Author(s) have copyright to the thesis, including the exclusive right to publish the document (The Copyright Act §2).
All theses fulfilling the requirements will be registered and published in Brage HiM, with the approval of the author(s).
Theses with a confidentiality agreement will not be published.

I/we hereby give Molde University College the right to, free of charge, make the thesis available for electronic publication: ☐**yes** ☒**no**

Is there an agreement of confidentiality? ☒**yes** ☐**no**
(A supplementary confidentiality agreement must be filled in)
- If yes: **Can the thesis be online published when the period of confidentiality is expired?** ☒**yes** ☐**no**

Date: 25.06.2015

Høgskolen i Molde

STANDARD AGREEMENT

This agreement is between

Aliaksei Liasnoi ....(Student(s)).

Irina Gribkovskaia .... (Faculty Advisor at Molde University College),

Statoil .... (Company/Institution),

And Molde University College/MSc Logistics Program Coordinator, concerning the use of specifications and results reported in the Master's degree thesis in accordance with the study plan for the Master's degree program in Logistics at Molde University College.

1. The student will complete the work assigned for the Master's degree thesis in cooperation with the company/institution
(Optional):..

The title of the thesis is: Periodic supply vessel planning problem with flexible departures and coupled vessels

2. The student has copyrights to the thesis. Those copies of the thesis submitted for evaluation along with descriptions and models, such as computer software that is included as part of or as an attachment to the thesis, belongs to Molde University College. The thesis and its attachments can be used by the College for teaching and research purposes without charge. The thesis and its attachments must not be used for other purposes.

3. The student has the right to publish the thesis, or parts of it, as an independent study or as part of a larger work, or in popularized form in any publication.

4. The company/institution has the right to receive a copy of the thesis with attachments, and the College's evaluation of it. The company/institution will have three (3) months from the time the thesis is submitted to the College for censoring to determine whether a patent is possible and to apply for a patent for all or part of the results in the thesis. The specifications and results in the thesis can be used by the company/institution in its own activities.

5. An additional confidentiality agreement may be entered into between the parties, if the company/institution sees this as necessary.

6. Each part of the agreement should have one copy of the agreement. The final signature should be from the Program Coordinator/Dean validating the agreement.

Place: Molde

Date of final signature: 12.06.2015

Student(s)

Irina Gribkovskaia
Faculty Advisor

Ellen K. Norlund
Company/Institution

Aild Hoff
Program Coordinator / Dean

Høgskolen i Molde

SUPPLEMENTARY AGREEMENT/CONFIDENTIALITY AGREEMENT

(Supplement to Point 5 of the Standard Agreement)

This agreement is between

Aliaksei Liasnoi ....(Student(s)).

Irina Gribkovskaia. (Faculty Advisor at Molde University College),

Statoil .... (Company/Institution),

and Molde University College/MSc logistics Program Coordinator. This is a supplement to the Standard Agreement, which defines the use of specifications and results reported in a Master's degree thesis in accordance with the study plan for the Master's degree program in Logistics at Molde University College.

1. As stated in Point 5 of the Standard Agreement the parties named above agree to limit access to the Master's degree thesis for 5.. Year(s) from the date of this agreement. The limit should be 1 to 5 years, for longer confidentiality periods, contact the program coordinator for a special agreement.

2. The reason for keeping the thesis results confidential is so that the company/institution can avoid giving away information that would give others a competitive advantage.

3. The thesis must be submitted to Molde University College. It will be kept in a secure room in the main archive at the College during the period agreed upon. The thesis can be borrowed during this period if a written request is submitted and the company/institution gives permission. Academic staff members can access the thesis after signing a declaration of secrecy. After the period of confidentiality is ended the thesis will be placed in the library for general use.

4. Four copies of this agreement must be signed so that each party will have a copy. The agreement is valid when it has been approved and signed by the program coordinator or Dean at Molde University College.

5. The confidentiality agreement does not exclude members of the academic staff from being present at the final oral presentation of the thesis.

Place...Molde...

Date of last signature...12.06.2015...

Student(s)

Irina Gribkovskaia

Faculty Advisor

Ellen K. Norlund

Company/Institution

Agreement approved by:

Arild Hoff

Program Coordinator/Dean

Abstract

The periodic supply vessel planning problem arises in offshore oil and gas operations. Supply vessels provide offshore installations with necessary supplies on periodic basis from offshore supply base according to weekly schedules. This thesis proposes a large neighbourhood search heuristic for a real-world periodic supply vessel planning problem faced by Statoil ASA. Formerly, that problem was grounded on fixed departure times for voyages during the day. However, we examine the effectiveness of flexible departure times for voyages during the day. As well as we extend vessels utilization up to two weeks in order to introduce the concept of coupled vessels, i.e. vessels, which can sail alternating routes on consecutive week. On small and medium problem instances, where the optimal solutions are known, the heuristic always finds optimal and near-optimal solutions. The heuristic is also capable of solving problems of large size within reasonable time, where the optimal solutions are unavailable.

Keywords: Maritime transportation, periodic routing, offshore upstream logistics, routing and scheduling, decision support system, large neighbourhood search heuristic, flexible departures, coupled vessels.

Acknowledgment

Immeasurable appreciation and deepest gratitude for the help and support are extended to many persons who in one or another way have contributed in making this study possible. I would like to extend my sincere thanks to all of them.

First and foremost I offer my sincerest gratitude to my supervisor, professor Irina Gribkovskaia, for her advices, guidance, valuable comments, suggestions and provisions. Without her incredible support and timely wisdom, this thesis work would have been a frustrating and overwhelming pursuit.

I would like to express my special gratitude towards to PhD candidate Yauheni Kisialiou for his valuable input, general assistance and practical advices.

I am highly indebted to Ellen Karoline Norlund from Statoil and her co-workers for providing necessary information for this research and valuable feedback.

I would like to thank Aliaksandr Shyshou for providing his insights, helpful comments and recommendations.

I am grateful to Molde University College for giving the opportunities to study in Norway during two years on the MSc program in logistics. I will never forget my days in Norway at the Molde University College.

Especially, I would like to thank the members of my family and my friends for the encouragement which helped me in completing this thesis, they served as my inspiration to pursue this undertaking and they fully supported me at the time when it was really needed.

Aliaksei Liasnoi

Molde, Norway

August 2015

Contents

List of Figures

List of Tables

Abbreviations

LNS large neighbourhood search. 10, 25, 30–32, 36, 38, 41, 42, 57, 58, 70, 71, 74, 75, 78, 79, 82, 86

PSV platform supply vessel. 18–23, 28, 36, 42, 45, 57, 58, 74

PSVPP periodic supply vessel planning problem. 10, 17, 18, 20, 24, 25, 27, 29–31, 38, 42, 58, 67, 85, 87

TSP travelling salesman problem. 28, 38, 39

TSPMTW travelling salesman problem with multiple time windows. 28, 38, 39

VBF voyage-based formulation. 24, 25

1 Introduction

The petroleum industry plays a vital role in energy supply and economic development of many different countries. Oil and gas are produced offshore in Norway. Offshore petroleum operations (production and drilling) are associated with high costs and risks; hence, continuous production of oil and gas is crucial for offshore installation. In order to provide efficient production, offshore facilities require regular supplies from onshore supply base, any delay in delivery may lead to production stop and loss of money. Special supply vessels are used to deliver necessary commodities on regular basis. The cost of transportation is very high as supply vessels are very expensive. It means that supply vessels have to be efficiently utilized and well thoughtful planning has to be made in order to provide cost-efficient and high quality supply service.

The supply vessel planning is an actual problem, which takes its roots in real-world applications, and usually it's tied up with high expenses accompanied by high service level. Supply vessel activities, which were not planned properly may require the need of extra vessel or helicopter to deliver supplies inducing additional costs. This underlines the significant importance of supply vessel planning and shows that the vessel plans should ensure that service is provided as required at minimal cost. Thus, the practical interest of the thesis shows that the development of methods that are able to generate optimal or nearly optimal supply vessel plans is of high importance for offshore operators and in general for the entire oil industry nowadays. From the scientific point of view, the supply vessel planning is compound and complicated. It represents a combinatorial optimization problem, which involves simultaneous decision-making on fleet sizing, routing and scheduling. Moreover, for problems of large sizes, a creation of the complex algorithm is required, that makes the problem even more challenging. Furthermore, the planning problem is a huge dimension task, the number of offshore installations increases all time together with expansion of the Norwegian petroleum industry. The total growth in discovered resources in 2013 has been estimated at 114 million Sm^3 o.e. Twenty new discoveries were made in 45 exploration wells (The Ministry of Petroleum and Energy and Norwegian Petroleum Directorate 2014).

In this research we consider a periodic supply vessel planning problem rooted in real-world application faced by Statoil ASA, the largest Norwegian offshore oil and gas operator. The problem is considered at the tactical level, where the planning horizon of supply services is one week. The problem consists of determining which vessels to use, their corresponding routes and schedules during the planning horizon. The planning involves such decision-making as: allocation of voyages to vessels (packing problem), sequencing of installations visits on a single voyage (routing problem), scheduling departure times for voyages of single vessel (scheduling problem). Each of them represents an NP-hard combinatorial problem, and taken together they make the problem extremely hard to solve. For the problems of such a size exact methods would not be able to provide optimal solution in reasonable time. Real-life instances of the problem are usually of a large size, involving over 20 installations to be served in a week, the required number of visits for each installation might be from 1 to 5, with many vessels required.

There is a limited number of literature on such problem nowadays. Moreover existing solution methods for tactical supply vessel planning are developed for a simplified variant of the problem, where vessel departure times from supply base are fixed within a day. This assumption has its practical explanation related to the opening hours at supply base, however it may lead to inefficient solutions and not optimal use of resources.

The purpose of the master thesis is to develop a decision support tool for a real-world single base periodic supply vessel planning problem with flexible departures and coupled vessels faced by Statoil ASA which is able to provide a new supply pattern for Mongstad supply base.

The main focus of the work will be on the development of a metaheuristic algorithms, which are able to yield accurate and relatively good solutions for real-life large size problem instances in reasonable time. As well as we set a goal to provide solutions which shows better results than known heuristics approaches. The metaheuristic algorithm will be validated on small size instances with the help of the two-phase method. It is based on voyage generation algorithm at the first stage and set-covering model at the second stage. Developed algorithms will be compared to known metaheuristic algorithms with fixed departures on medium and large size instances.

The structure of the thesis is the following:

In Chapter 2, we give a description of a planning problem, identifying constraints and objectives. In Chapter 3, we perform related literature review, together with an analysis of what is missing in existing solutions in comparison with formulated problem. In Chapter 4, we give an overview on methods that would be further extended to be applicable to formulated problem. In Chapter 5 we describe the research objective of the thesis related to the algorithms development, validation and analysis. Chapter 6 presents a detailed description of algorithms developed especially for construction of supply vessels schedules with flexible departures and coupled vessels. Chapter 7 contains a description of experiments setup, test instances and the comparative analysis of results for different problem sizes. In Chapters 8, we give a conclusions and provide directions for further research. Bibliography and appendices completes the thesis.

2 Problem Description

The research conducted in this thesis represents a real-life problem of Statoil ASA, the largest Norwegian oil and gas operator on the Norwegian continental shelf with approximately 22500 employees and total revenue of NOK 622,7 billions. Besides the Norwegian continental shelf Statoil has ongoing development and production in 11 countries: Algeria, Angola, Azerbaijan, Brazil, Canada, Libya, Nigeria, Russia, the UK, the US, and Venezuela (Statoil ASA 2015). In this research we focus on a particular problem of a single supply base in Mongstad among all the other onshore supply bases located along the Norwegian continental shelf, which Statoil operates nowadays.

In this section a problem, considered in this thesis, is formulated as a variant of the periodic supply vessel planning problem (PSVPP) for a single supply base along with its characteristics and the solution (weekly sailing plan) is illustrated in the example.

2.1 Periodic supply vessel planning

The periodic supply vessel planning problem (PSVPP) concerns how to build a least cost schedule of a given planning horizon for supply vessels that serves offshore installations from a supply base. The PSVPP consists of taking simultaneous decisions on determining the optimal fleet composition of vessels needed to perform a supply operations to offshore installations from onshore supply base together with vessels routes and schedules. In relation to the planing horizon, Christiansen et al. (2007) distinguish three levels of maritime transportation planning: strategic planning, tactical planning and operational planning. Strategic planning is applied to a long-term decisions with a time frame greater than one year. It involves market and trade selection, fleet size and mix decisions (type, size, and number of vessels), network and transportation system design. It aims to maximize the service quality with different budget restrictions or to minimize total costs meeting the service requirements. Tactical planning refers to such decisions as adjustments of a fleet size, vessels routing and scheduling, inventory management and berth scheduling. The planning horizon on a tactical level is usually considered anywhere between one week and a year. Operational planning refers to day-to-day decisions. It concerns operations on the particular voyage, such as vessels speed selection, vessels loading and unloading operations at the supply base or at the offshore installation and as well as environmental routing decisions on weather conditions and ocean currents.

The planning horizon for the schedule, in this particular problem, is considered to be one week (tactical planning). Such schedule is repeated for several weeks or months until there is a need to make some changes and adapt the current schedule. Such circumstances to update the plan might be: new installations to be serviced, changes in the number of visits or time windows for some installations or even major demand changes. Moreover, scheduling should be done in such a way to insure a fairly spread departures to each installation throughout a week. The reasons to do so hides behind the offshore installations needs, which require even supply throughout a week. To summarize different aspects of the PSVPP in our case, which are relevant to consider before making a sailing plan, let's take a closer look at supply base, offshore installation, supply vessels and other important characteristics.

2.1.1 Supply base

The onshore supply base serves as the starting point for platform supply vessel (PSV) for loading/unloading cargo operations. A cargo has to be delivered or discharged to or from the offshore oil and gas installations. Supply base servicing a number of offshore installations and it has a limited number of PSVs available to perform these operations. As well as there are several constraints and assumptions that have to be considered for a supply base. A limited number of berths and personnel availability implies a limited number of PSVs to be serviced simultaneously, that leads to a limited base capacity. Supply base may have specific opening hours, i.e time windows constraint, when it is possible to perform service, in Norway working hours are usually considered to be from 8:00 to 16:00. The turnaround time for PSVs at the supply base is estimated to be about 8 hours, which is used to perform the loading and unloading operations. The departure time for PSVs is dependent on working hours of the supply base and assumed to be fixed throughout the day (at 16:00).

2.1.2 Offshore installations

The offshore installations performs main operations for oil and gas production, each offshore installation may require different number of visits per week, usually the one performing drilling operations are more demand and visit intensive than the one performing oil and gas extraction. Hence, each offshore installation weekly requirements are represented by a number of visits per week and a demand (volume of cargo) to be delivered in order to satisfy their needs. The demand between visits is uniformly distributed. It is assumed that spread of visits throughout a week is not considered directly, but it's more important to consider the spread of PSVs departures instead. This set up a rule for the offshore installations to submit a demand request for the upcoming visit before the PSV leaves the supply base at a given day. Additionally each offshore installation has its opening hours, when the service can be provided, it may differ from 24/7 for drilling installations or from 7:00 till 19:00 for production platforms.

2.1.3 Supply vessels

The platform supply vessel (PSV) performs transportation of commodities to and from offshore installation. Considering the supply vessel fleet, each PSV may have different capacity limits for cargo transportation as well as different sailing speeds. For instance, different capacities indicates that some PSVs may not be able to sail some voyages where the total demand exceeds vessel's capacity. Supply vessel costs are characterized by a charter cost for a weekly usage, as well as variable fuel cost, which is dependent on fuel consumption and vessel speed. For instance fuel consumption may differ while performing different operation such as: loading/unloading at the supply base, sailing to the destination platform or loading/unloading at the offshore installation.

2.1.4 Routes and voyages

The route is defined as a collection of voyages sailed by specific PSV during the week. By the voyage we understand a sequence of installation visits by particular PSV starting and ending at the supply base. Each voyage has a minimum and maximum duration in days, which is specified by lead-time delivery requirements from installation, it commonly lasts for 2 or 3 days. There is also a minimum and maximum requirement on the number of installations to be visited in one voyage and it is from 1 up to 7 installations respectively. Furthermore, it should be guaranteed that there is no overlap between voyages of the same vessel in the schedule.

2.1.5 Objective

The objective of the PSVPP is to build a tactical sailing plan that will eventually minimize the sum of vessel charter costs and fuel costs during sailing, servicing and waiting. In order to do that we need to take simultaneous decisions on determining the number of PSVs required to perform the supply service, the type of PSVs and their weekly voyages, which are described by the sequence of installations visits. Additionally, the weekly route plan should guarantee a fairly spread of departures to each installation throughout a week.

2.1.6 Weekly sailing plan

Supply vessels schedule is represented by a set of sequential voyages assigned to the days of the planning horizon. The Figure 2.1 shows an example of weekly sailing plan. It illustrates daily schedules for three vessels, which are listed in the first column. From the supply base perspective it comprises three vessels and five offshore installations. Since onshore supply base usually has open hours from 8:00 to 16:00, this time window constraint should be satisfied when performing vessel servicing. Assuming that such service usually takes around 8 hours turnaround time (highlighted in dark green color), it should start at 8 o'clock in the morning to be completed before vessel departure at 4 o'clock in the evening. From the PSV perspective, weekly route is presented by a number voyages, displaying a sequence of offshore installations visits starting and ending at the supply base for each voyage,

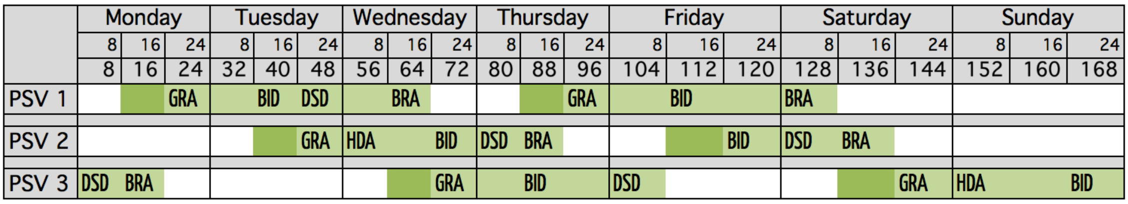


Figure 2.1: An example of weekly sailing plan

as well as departure time for each voyage (i.e. the first voyage of PSV 1 is: GRA-BID-DSD- starting at 16:00 on Monday). Numbers below each day of the week corresponds to the end of the eight-hour interval. The voyages have a colored outline on the Figure 2.1. In this example each vessel performs a schedule with two voyages during the week, while a total duration of voyage is limited up to three days.

2.2 Extensions of the problem

In order to provide a complete picture, it's necessary to put to the question some features of the problem described above. A consideration of PSV's departure times to be fixed has been dictated by supply base opening hours. However, in theory, it make sense to consider a 24-hours open period for the supply base, so that the dependency for PSV's departure times is relaxed and it is possible to look for a situations where it is beneficial to use different departure times. Thus, it has a meaning to consider the time of departures as an option, which can take different values as desired by the company and as a result can provide the opportunity to find a better solution. Such solution may have advantages of cost reduction for both cost components: for the vessel charter costs, when the vessel fleet size is reduced and for fuel costs, when a PSV spends less waiting time at the installation.

We call such relaxation as a allowance of flexible departures times and it means that any PSV has a possibility to depart in different points of time during the day. Hence, the algorithm should take care of selecting such departure times for each voyage, so that the cost of the solution is minimized. The Figure 2.2 shows the simplest example with the benefit of flexible departure times, where it is allowed to depart not only at 16:00, but at 8:00 as well. The PSV 2 has only one voyage (highlighted in purple color), which starts at 16:00 on Thursday, but since now we have a possibility to start voyages at 8:00 as well, the voyage could be easily moved to the other vessel (PSV 1) without violation of the constraints, so there is no longer a need of PSV 2 and the whole supply service could be provided with a use of only two supply vessels.

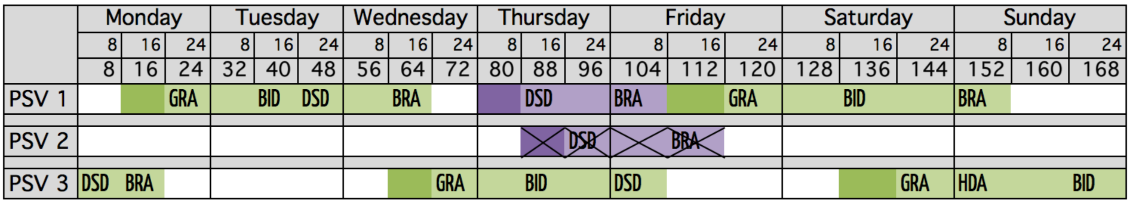


Figure 2.2: An example of weekly sailing plan with flexible departures

Since every voyage commonly lasts for 2 or 3 days, we may end up with so-called "end-of-week" effect situation, when the last voyage starts at the end of the week and ends at the begining of the next week. It may also happen that the first and the last voyages of the same vessel may overlap, that is not allowed when vessels are utilized for one week. However, if we extend the utilization period for PSVs up to two weeks, we may avoid this infeasibility by swapping voyages of any two PSVs on consecutive week, while having overlapping voyages in case of "end-of-week" effect. Thus, we introduce the concept of coupled vessels. By coupled vessels, we mean two or more PSVs which can sail each other weekly routes, servicing the same set of offshore installations. It's also guaranteed that coupled vessels have a necessary deck capacity to sail each other routes.

The benefit of coupled vessels is shown on Figure 2.3, where the PSV 1 have a total routes duration of 8 days on the first week and total duration of 6 days on the next week. PSV 1 sails alternating weekly routes with PSV 3, while PSV 2 repeats same routes on the second week. In theory, another benefit of coupled vessels could be the fleet size reduction, when the PSV has a single voyage that could be incorporated into another PSV's weekly route, so that first and last voyages of the vessel are overlapped. Then it will be possible to get rid of unused PSV and provide a schedule with coupled vessels for a reduced number of PSVs.

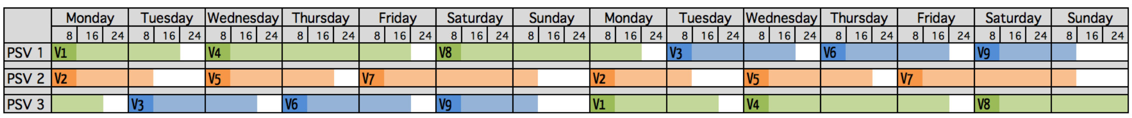


Figure 2.3: An example of weekly sailing plan with coupled vessels

To summarize, the problem described above will be extended with assumptions, that the PSVs departure times are flexible throughout the day. We assume possible departure time options to be 8:00 and 16:00. This assumption may give an opportunity to find a better solution compared to a fixed departure time at 16:00. For a large size problem instances, flexible departures ensure that many vessels may start from the base during the day. Additionally, we assume that PSVs are utilized for two weeks, allowing PSVs to sail alternating weekly routes in case of “end-of-week” effect with overlapping voyages.

3 Literature Review

Since PSVPP is related to periodic vehicle routing and scheduling, the problem can be classified as fleet sizing and periodic vehicle routing and scheduling. There are some noteworthy articles and scientific papers by Fagerholt et al. (2000), Halvorsen-Weare et al. (2011) and Halvorsen-Weare et al. (2012), Shyshou et al. (2012), Norlund et al. (2013).

Fagerholt et al. (2000) considered the problem of determination of efficient policy for a supply operation in the Norwegian Sea. The authors evaluated scenarios of having some installation closed for service during the night. For this they developed a two-phase approach where on the first stage all feasible routes are generated. On the second stage vessels to be used and weekly schedules are determined by solving set-partitioning problem. The authors consider simplified version of our problem, only assignment problem of voyages to vessels was made, there is no scheduling and no spread of departures included into the study.

Halvorsen-Weare et al. (2012) considered periodic supply vessel problem with some complicating aspects. The authors presented a two-phase voyage-based formulation (VBF) that took into account service capacity of the supply base, maximum and minimum route duration, and spread of departures of supply vessels. At the first phase all possible cheapest sequences for voyages are generated. Later, at the second phase, those sequences are used as an input to a voyage-based set-covering model with numerous side constraints. As well the authors dealt with the problem of weather uncertainty that may cause delays to planned voyages and schedules. They dealt with this problem by creating a robust voyages schedule requiring slacks to each voyage duration. The authors have tested only small and medium instances.

The approach to solve the PSVPP, proposed by Halvorsen-Weare et al. (2012) can find a solution for a fixed departure times and when the number of installations is less then 12, but when it comes to a large number of installations, algorithm becomes intractable in a reasonable time.

Shyshou et al. (2012) proposes to use a large neighbourhood search (LNS) heuristic aimed to solve similar problem. The LNS heuristic is aimed to solve difficult planning problem involving fleet composition and vessel scheduling decisions. It finds optimal or near-optimal solutions, repeatedly improving the current solution by looking for a better solution, which is in the neighborhood of the current solution. In that sense, the neighborhood of the current solution includes a possibly large number of solutions. In a huge search area you can have the following behavior where early decisions are never reconsidered and you spend time searching in only one region of the search space. With LNS when you realize that you are stuck for too long, you perform a restart and visit another place to improve the current best solution. Thus, at each iteration a large portion of the solution is rearranged and a broader exploration of the search area is performed.

Comparing with a two-phase approach (Halvorsen-Weare et al. 2012), the LNS performs better when it comes to a large number of offshore installations (more than 12). The authors showed the ability of heuristic to find solutions for big instances with 31 installations.

Halvorsen-Weare et al. (2011) presented an approach to build a robust schedule for the supply vessel tactical planning problem. Authors presented a three-phase approach that combines optimization and also simulation to ensure robust schedules. At the first phase all candidate voyages the vessels may sail are generated. At the second phase, they simulate candidate voyages to assign a robustness measure. And at the last stage the VBF model including robustness measure is solved.

Norlund et al. (2013) examined minimization of supply vessels emissions by optimizing sailing speeds. The authors applied two-phase approach to solve the problem. For emissions reduction they introduced several speed optimizing strategies that were used in construction of periodic vessels schedules. The strategies used voyages inter and intra waiting time during voyage generation procedure. Application of such strategies yielded 25% in the reduction of fuel consumption without increasing the fleet size.

The PSVPP described in the above articles and scientific papers consider a planning horizon of only one week. Developed approaches provide solutions only for small or medium size of instances, except Shyshou et al. (2012). Furthermore, supply vessels departure time from supply base was considered to be always fixed.

None of the methods above can be applied directly for a problem with flexible departures and coupled vessels. We can conclude that new methods should be developed for the problem studied in this thesis. Due to the complexity of the problem and large size of real-life problem instances, it seems that it's reasonable to develop a metaheuristic algorithm for the problem, which is able to provide a relatively good solutions in reasonable time.

4 Methodology

In this section we describe two existing approaches which were applied to solve the PSVPP with fixed departure times and without consideration of coupled vessels schedules, as well as we bring their advantages and disadvantages. It is also discussed, how these approaches can be extended or modified for the purpose of our study, namely for the development of algorithms for the PSVPP with fixed departure times and coupled vessels.

4.1 Two-phase method

Two-phase approach was introduced by Halvorsen-Weare et al. (2012). The first phase involves voyage generation procedure, where all feasible candidate voyages are generated, on the second phase generated voyages are used as input to voyage-based model. The Figure 4.1 provide a schematic overview of the two phase approach.

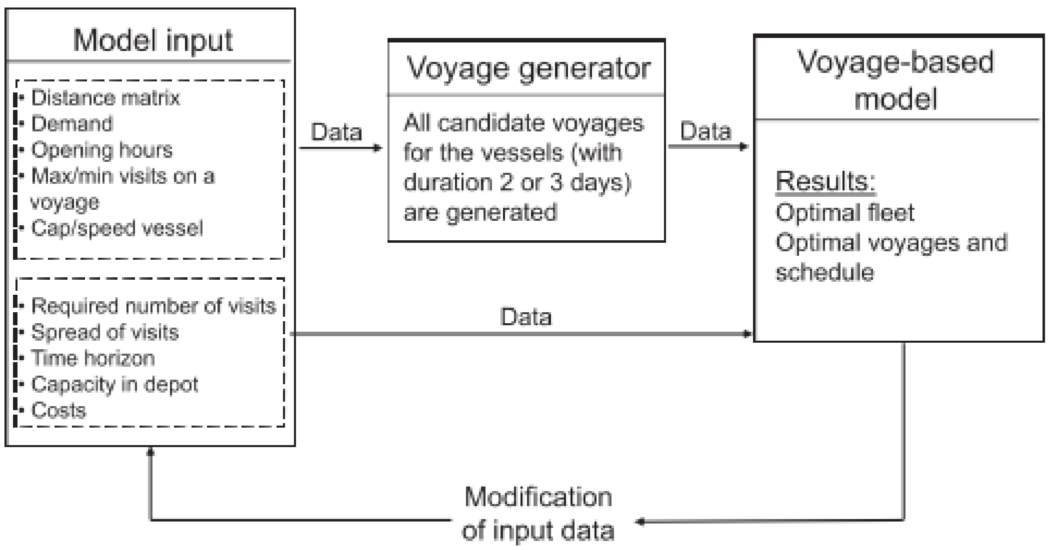


Figure 4.1: A voyage-based method (Halvorsen-Weare et al. 2012)

4.1.1 Voyage generation

For voyage generation, the authors first define all possible subsets of offshore installations that may be serviced by a given supply vessel. The size of the subset is restricted by the minimum and maximum number of installation on a voyage and PSV deck capacity. Then, for each subset of voyages, where at least one installation have opening hours travelling salesman problem with multiple time windows (TSPMTW) is solved. For subsets of voyages that do not contain installations with time windows a standard travelling salesman problem (TSP) is solved. The output of the voyage generation procedure represent a set of all candidate voyages that is used as input to set-covering model. The pseudo-code of the voyage generation is presented in the Procedure 4.1.1.

Procedure 4.1.1 Generate voyages (Halvorsen-Weare et al. 2012)

```
create sets of vessels (VesselSet) with equal sailing speed
for all VesselSet do
    find vessel in VesselSet with largest loading capacity, vesselMax
    enumerate all sets of installations (InstallationSet) that fulfill minimum
    and maximum requirements on number of installations in a voyage and
    that does not exceed the capacity of vesselMax
    for all InstallationSet do
        find a voyage by solving a traveling salesman problem with time win-
        dows starting and ending at the supply depot where all installations in
        InstallationSet are visited exactly once
        if voyage does not violate minimum and maximum duration then
            add voyage to VoyageSet for vesselMax (VoyageSet[vesselMax])
        end if
    end for
    for all vessels in VesselSet not vesselMax do
        for all voyages in VoyageSet[vesselMax] do
            if voyage does not violate capacity of vessel then
                add voyage to VoyageSet[vessel]
            end if
        end for
    end for
end for
return all VoyageSets
```

4.1.2 Voyage-based model

The voyage-based model formulation solves the PSVPP with the objective of minimizing the fleet size and fuel cost.

Let V be the set of supply vessels and N is the set of all offshore installations. Let R_j be the set of all voyages vessel $j \in V$ may sail. T is the number of days in the week and L is the set of all possible voyage durations in days. Then set R_{jl} is the set of all voyages of duration $l \in L$ vessel $j \in V$ may sail. c_j^{TC} is a weekly vessel charter cost for vessel $j \in V$. c_{jk}^S represents sailing and service cost for voyage $k \in R_j$ sailing by vessel $j \in V$, f_j denotes the number of days vessel j is available during the week, s_i is the number of visits required by installation $i \in N$ during the week and b_t is the maximum number of supply vessels that may be serviced at the supply base on day $t \in T$. Parameter d_{jk} represents, rounded up to the nearest integer, duration of a voyage k sailed by vessel j ($j \in V$, $k \in R_j$). A special parameter $0 \leq h_r \leq |T|$ representing sub horizon for the installation with visit frequency $r \in F$ is defined to control the spread of departures during the planning horizon. In addition we define two parameters $\underline{p}_r$ and $\overline{p}_r$ representing minimum and maximum number of visits for a n installation during sub horizon h_r. a_{ijk} is a binary parameter that equal 1 if vessel j visits installation i on voyage k. And finally the following decision variables are used: y_j is 1 if vessel j is used, 0 otherwise. And x_{jkt} is 1 if vessel j sails voyage k starting on day t, 0 otherwise.

$$\min \sum_{j \in V} c_j^{TC} y_j + \sum_{j \in V} \sum_{k \in R_j} \sum_{t \in T} c_{jk}^S x_{jkt} \tag{4.1}$$

subject to

$$\sum_{j \in V} \sum_{k \in R_j} \sum_{t \in T} a_{ijk} x_{jkt} \geq s_i, i \in N \tag{4.2}$$

$$\sum_{k \in R_j} \sum_{t \in T} d_{jk} x_{jkt} - f_j y_j \leq 0, j \in V \tag{4.3}$$

$$\sum_{j \in V} \sum_{k \in R_j} x_{jkt} - f_j y_j \leq b_t, t \in T \tag{4.4}$$

$$\sum_{k \in R_{jl}} x_{jkt} + \sum_{k \in R_j} \sum_{q=1}^{l-1} x_{jk,(t+q) \mod |T|} \leq y_j, j \in V, t \in T, l \in L \tag{4.5}$$

$$\underline{p}_r \leq \sum_{j \in V} \sum_{k \in R_j} \sum_{h=0}^{h_r} a_{ijk} x_{jk,(t+h) \mod |T|} \leq \overline{p}_r, i \in N_r, t \in T, r \in F \tag{4.6}$$

$$y_j \in \{0,1\}, j \in V \tag{4.7}$$

$$x_{jkt} \in \{0,1\}, j \in V, k \in R_j, t \in T. \tag{4.8}$$

The objective function (4.1) states that total vessels charter and fuel cost must be minimized. The charter cost is much higher that the fuel cost associated with sailing and servicing at an installation, the primary objective is to define the most cost effective fleet composition. Constraints (4.2) ensure that each installation receives the required number of visits during the planning horizon. Constraints (4.3) ensure that total duration of voyages sailed by a vessel does not exceeds the maximum number of days a vessel is available. Constraints (4.4) state that the number of vessels serviced at the supply base on day t should not exceed its maximum. Constraints (4.5) ensure that a vessel does not start on a new voyage before it returned to the supply base from the previous. Constraint (4.6) ensure even spread of departures to each installation during the planning horizon. And finally, constraints (4.7) and (4.8) impose binary requirements on the variables.

4.2 Large neighbourhood search algorithm

4.2.1 Heuristic summary

The large neighbourhood search (LNS) metaheuristic was initially proposed by Shaw (1998) to solve vehicle routing problems. In this section we describe the LNS metaheuristic adopted for the PSVPP provided by Shyshou et al. (2012). The algorithm is applied for a number of restarts, which is initially specified by the user. At each restart a feasible initial solution is generated. Then, for a given number of iterations, a transition to a neighbourhood solution is made and the solution is tried to be improved. A transition to a neighbourhood solution is provided by *Remove Visits* and *Insert Visits* procedures. The first procedure removes some visits from several voyages and puts them to a pool S of not inserted visits and the second procedure tries to insert them to other voyages. When feasible solution with the best relocation of removed visits is defined, a set of improvement procedures is applied while the cost of the solution decreases. Then a post improvement procedure is performed with the aim to reduce the fleet size and the next iteration is started. The post-improvement procedure may result in the reduction of the number of voyages below the level when a feasible solution can be generated at the next iteration. To eliminate such infeasibility, at the beginning of the next iteration empty voyages are created. Creation of empty voyages is performed if only the number of voyages drops below a certain minimum. For this purpose the lower bound on the number of voyages is calculated before running the algorithm. Conceptual flowchart of the LNS heuristic is provided on the Figure 4.2

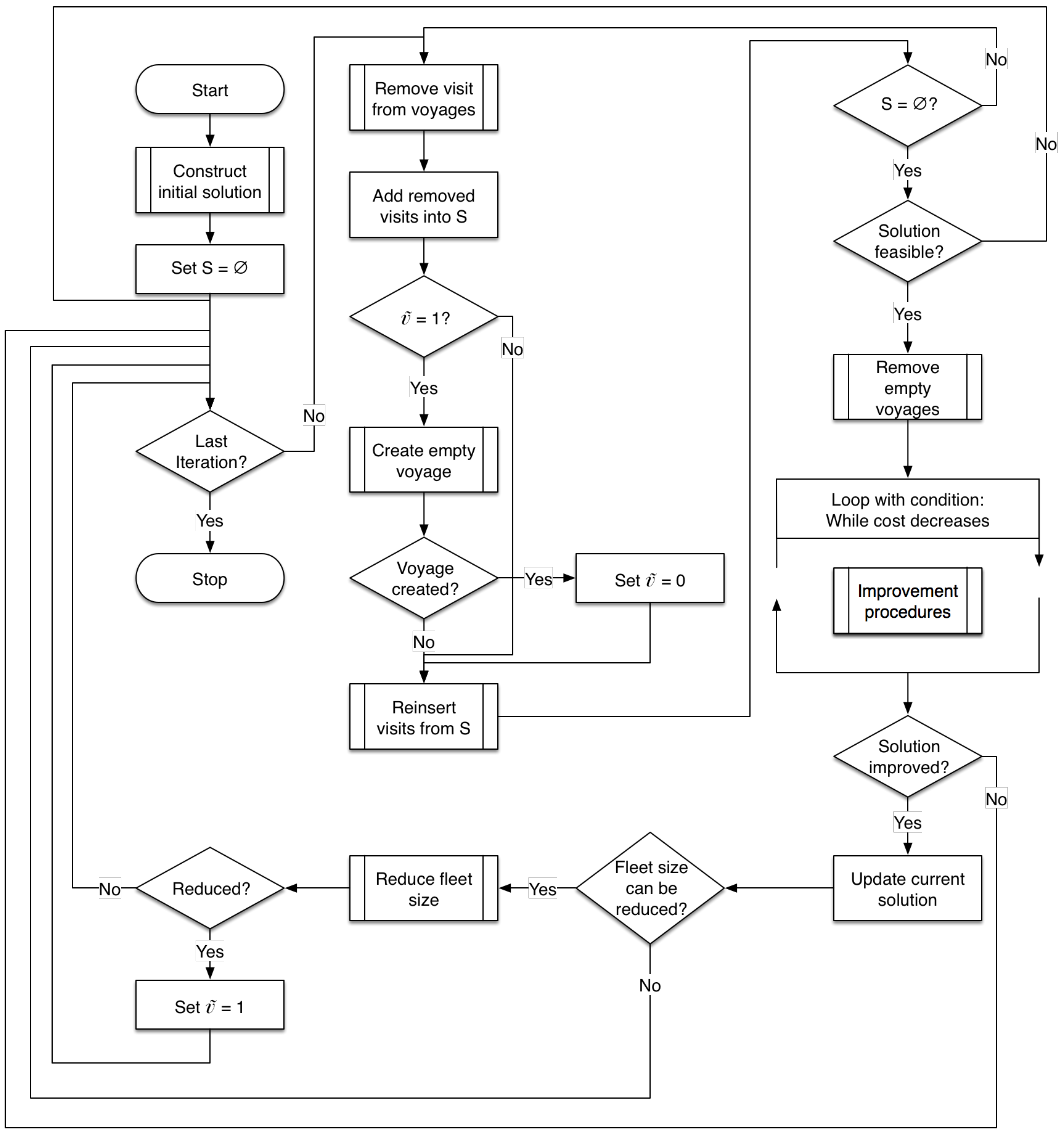


Figure 4.2: A flowchart of the LNS algorithm for the PSVPP

4.2.2 Construction of initial solution

Initial solution represents a collection of voyages and supply vessels to which voyages are assigned. At each restart initial feasible solution is generated. For each installation a feasible pattern of departure days is randomly generated based on supply base capacity and required number of visits. Feasible departure patterns are pre-generated and chosen randomly. Thus, for each day we define a set of installations to which a vessel is to departure for servicing them. Then, for each day constructed voyages with respect to constraints on minimum and maximum number of visits per voyage, deck capacity and vessel availability etc. Constructed voyages are then assigned to vessels or a vessel. The procedure is repeated until some feasible solution is found which is then used further in the LNS iterations.

4.2.3 The LNS iteration

The heuristic is applied for a given number of restarts. At each restart an initial feasible solution is constructed and further a number of the LNS iterations is performed. After initial solution is generated, a random number of visits is removed from several voyages and is put to a pool S of uninserted visits. If there are uninserted visits, remained from previous iteration (remained after post improvement procedure at the end of iteration) and the number of voyages is below predefined minimum then empty voyages are created. The logic lying behind empty voyages creation is twofold. On the one hand it eliminates possible schedule infeasibility and on the other hand creation of empty voyages only if the number of voyages drops below a certain minimum prevents creation of larger number of voyages with few visits. Further an attempt is made to insert removed visits to voyages using regret-like heuristic. If after a certain span of time or number of attempts there are no any voyages for insertion the next LNS iteration is performed.

As soon as set S becomes empty (all removed visits were inserted to voyages), local improvement procedures are applied with the aim to find more cost efficient schedule. First, an attempt to reduce the number of voyages is made. Visits of the shortest voyages are removed and tried to be reassigned to other voyages. After reducing the number of voyages, the remained voyages are reassigned to vessels with the aim to reduce the number of used vessels. Then an attempt is made to reduce the total duration of all voyages. The logic of the procedure is to increase vessel's idle time and further reassign voyages to vessels again, for proper packing, to make one or more used vessels idle. And finally a procedure that relocates visits between voyages is applied to reduce total sailing and service cost.

Local improvements procedures are applied while the total costs decreases. After the local improvement stage and before going to the next iteration an attempt is made to reduce the fleet size again. Voyages of a vessel are partially reassigned to other vessels by reassigning visits of voyages. Not reassigned visits are put to the set of uninserted visits S and vessel is marked as "not used". At the beginning of the next iteration, after visits are removed from some voyages, if the number of voyages turns out to be below the predefined minimum (since at the previous iteration the number of voyages could be reduced after the post improvement attempt to reduce the fleet size) empty voyages are created to eliminate possible infeasibility. At the end of each iteration the solution is stored in case of its feasibility and when all iterations were performed the cheapest solution is returned.

4.2.4 Improvement procedures

This section describes improvement procedures performed after remove insert procedures at the beginning of the iteration.

Intra voyage optimization

The procedure is applied after introducing changes to a voyages, removing from or inserting to it visits. The aim of the procedure is to reduce the duration of a voyage. One visit is removed a time and first-accept rule applying reinserting this visit into the first position which leads to reduction of the voyage duration. The procedure is applied while improvements are found.

Reducing the number of voyages

The aim of the procedure is to reduce the total number of voyages for fuel cost reduction and for proper subsequent packing (or assignment) between vessels in the next improvement procedure – reduce the number of vessels. The procedure involves the following steps:

1. Select a voyage;
2. Remove one visit and insert it into another voyage;
3. Repeat this procedure for all visits of the voyage;
4. If all visits were removed accept the changes;
5. Go to step one.

The procedure is stopped when further reduction cannot be achieved is not or the number of voyages reached its predefined minimum.

Reassigning voyages to vessels

The procedure tries to better pack voyages between vessels to make the schedule as tight as possible. The aim is to reduce the number of used vessels. The following steep are performed:

1. Select a vessels;
2. Try to reassign voyages of the selected vessel to other vessels. The vessel to which a voyage is reassigned may not be currently in use, but it must be of smaller size than the selected;
3. Changes of the schedule come into force if only all voyages of vessel were reassigned.

Reduce total duration of voyages

The aim of the procedure is to reduce duration of voyages measured in days. The logic lying behind is a vessel is loaded at the supply base between 8.00 and 16.00 and must departure at 16.00. If a vessel arrives to the base later than 8.00 it may start a new voyage only on the next day, so even small reduction of the voyages duration in hours may lead to reduction of the duration in days. The steeps are as follows:

1. Select a voyages;
2. For each visit of a voyage evaluate all possible relocations. Duration of the destination voyage should not be increased;
3. Implement the best relocation in terms of:
    - The number of possible relocations for a visit;
    - Difference between the increase in time of the target voyage and decrease of the source voyage;
    - Increase of the schedule cost.

If the duration is reduced the reinsertion pattern is stored. After evaluations for all visits of all voyages are done, relocation yielding the smallest objective function increase is implemented. The procedure is performed while there are feasible visits relocations or voyage duration is decreased by one day.

Relocate visits to other voyages

The aim of the procedure is to reduce the total sailing cost. Relocation of all visits of all voyages is performed maintaining feasibility and while objective function improves.

5 Research Objectives

The main goal of this research is to develop a decision support tool for a real-world single base periodic supply vessel planning problem with flexible departures and coupled vessels. The research work includes the development of two algorithms providing a tactical schedule for one week planning horizon with flexible departures and one algorithm providing a tactical schedule with flexible departures and coupled vessels, when vessels are utilized for two weeks, so that PSVs are allowed to sail alternating weekly routes.

The first algorithm is based on a two-phase method, proposed by Halvorsen-Weare et al. (2012). At the first stage the algorithm of voyage generation with flexible departures will be developed and at the second stage the set-covering model will be used with appropriate modifications to incorporate flexible departure times.

The second algorithm is based on the idea of the large neighbourhood search (LNS) metaheuristic, proposed by Shyshou et al. (2012). The goal is to modify the algorithm in such a way that it will be allowed to build a supply vessel schedule with flexible departure times, so that the departure time during a day will become a user-defined option and it will be up to the algorithm to decide what departure time is better for each voyage.

The third algorithm will further extend the LNS metaheuristic with the concept of coupled vessels in addition to the flexible departure times. It will provide a schedule for the PSVs, where two or more vessels can sail each other weekly routes, servicing the same set of offshore installations.

We intend to create a new implementation of the metaheuristic algorithms in order to optimize the computational time and maintainability of the application. The algorithm should be able to find optimal or quasi-optimal solutions in less computational time, than the existing algorithm.

Comparative analysis of two-phase method with single and flexible departure times will be conducted in order to assess the performance of flexible departure times compared to fixed departure times. Subsequently, the comparative analysis of the two-phase method and heuristic algorithm with flexible departure times will be conducted on small and medium problem size instances in order to validate the developed metaheuristic algorithm with respect to the quality of solutions.

For a large problem instances, the comparative analysis of algorithms will be conducted with a purpose of identification of deviations of the solution from the best-found solution and corresponding computational time. As well as the analysis of the influence of the coupled vessels concept on the metaheuristic with flexible departures will be held to assess the contribution to the cost reduction. The metaheuristic algorithm will be tested on real-life instances as well with respect to its efficiency.

6 Solution Algorithms for the PSVPP with Flexible Departures and Coupled Vessels

In this section we provide modified two-phase method of Halvorsen-Weare et al. (2012) and the LNS heuristic by Shyshou et al. (2012) for the PSVPP with flexible departures and coupled vessels.

6.1 Two-phase method with flexible departures

6.1.1 Voyage generation recursive algorithm

The voyage generation algorithm is pretty similar to the one described in Section 4.1.1 and it is based on the principle of recursive call. The information of offshore installations and supply vessels are the inputs for the algorithm, as well as pre-generated distance matrix. The recursive procedure is called as much times as many departure times are specified by a user. The recursive algorithm starts with sequencing installations to build voyages. Starting from one installation and adding new installations to the sequence. It remembers the state of the sequence and if it's no longer possible to add new installation due to some constraints violation, algorithm returns back to the previously saved state and continues with remaining installations. The voyage size is limited by the minimum and maximum number of installations to visit and maximum capacity of the supply vessel. The algorithm calculates voyages sailing, waiting and servicing time, together they represents the total duration of the voyage. This duration is checked to lay between minimum and maximum voyage duration. In case of duration violation algorithm returns back to the previously saved state and tries to add another installations to the visits sequence. If all constraints are met, voyages are stored together with its cost. Then, for stored voyage a TSP or TSPMTW is solved, depending whether voyage contain installations with time windows or not. The output of generator represents a set of all candidate voyages that is used as input to set-covering model described in 6.1.2. The pseudo-code of the voyage generation is presented in the Algorithm 6.1.1.

Algorithm 6.1.1 Generate Voyages

create sets of vessels V with equal sailing speed and capacities;
for all vessels $j \in V$ **do**
  for all departure times $t \in T$ **do**
    recursively enumerate all sets of installations $i \in N$ that doesn't violate minimum and maximum number of installations in a voyage and vessel capacity;
  end for
  for all sets of installations $i \in N$ **do**
    if the set contains at least 1 installation with time windows **then**
      find a voyage k by solving TSPMTW;
    else
      find a voyage k by solving TSP;
    end if
    if voyage k does not violate minimum and maximum duration
      add capacity of the vessel **then**
      denote by R_{jk} a set of voyages k for vessels j;
      add voyage k to R_{jk} for specific vessel j;
    end if
  end for
end for
return R_{jk};

6.1.2 Set-covering model

Let V be the set of supply vessels and N is the set of all offshore installations. Let R_j be the set of all voyages vessel $j \in V$ may sail. We define T as the number of available departure times during the week and D as the set of days in the week. Then T_d is the set of departure times for each day $d \in D$ and L is defined as the number of all different voyage durations (resulted after the voyage generation algorithm and measured in time intervals equaling to $24/m$, where m is the number of departure times per day). Set R_{jl} is the set of all voyages of duration $l = 1, \dots, L$ vessel $j \in V$ may sail. c_j^{TC} is a weekly vessel charter cost for vessel $j \in V$. c_{jkt}^{S} represents sailing and service cost for voyage $k \in R_j$ sailing by vessel $j \in V$ starting at departure time t, f_j denotes the number of days vessel j is available during the week, s_i is the number of visits required by installation $i \in N$ during the week and b_d is the maximum number of supply vessels that may be serviced at the supply base on day $d \in D$. Parameter d_{jkt} represents, rounded up to the nearest integer, duration of a voyage k sailed by vessel j ($j \in V$, $k \in R_j$) and G_l represents voyage duration number l measured in time intervals equaling $24/m$. A special parameter $0 \leq h_r \leq |T|$ representing sub horizon for the installation with visit frequency $r \in F$ is defined to control the spread of departures during the planning horizon. In addition we define two parameters $\underline{p}_r$ and $\overline{p}_r$ representing minimum and maximum number of visits for an n installation during sub horizon h_r. a_{ijk} is a binary parameter that equal 1 if vessel j visits installation i on voyage k and 0 otherwise. And finally the following decision variables are used: y_j is 1 if vessel j is used, 0 otherwise. x_{jkt} is 1 if vessel j sails voyage k starting at departure time number t, 0 otherwise. z_{jkd} 1 if vessel j sails voyage k starting on day d, 0 otherwise.

$$\min \sum_{j \in V} c_j^{TC} y_j + \sum_{j \in V} \sum_{k \in R_j} \sum_{t=0}^{T-1} c_{jkt}^{S} x_{jkt} \tag{6.1}$$

subject to

$$\sum_{j \in V} \sum_{k \in R_j} \sum_{t=0}^{T-1} a_{ijk} x_{jkt} \geq s_i, i \in N \tag{6.2}$$

$$\sum_{k \in R_j} \sum_{t=0}^{T-1} d_{jk} x_{jkt} - f_j y_j \leq 0, j \in V \tag{6.3}$$

$$\sum_{j \in V} \sum_{k \in R_j} \sum_{t \in T_d} x_{jkt} \leq b_d, d \in D \tag{6.4}$$

$$\begin{aligned} \sum_{k \in R_{jl}} x_{jkt} + \sum_{k \in R_j} \sum_{q=1}^{G_l} x_{jk,(t+q) \mod |T|} \leq y_j + 2(1 - \sum_{k \in R_{lj}} x_{jkt}), \\ j \in V, t = 0, \ldots, T-1, l = 0, \ldots, L \end{aligned} \tag{6.5}$$

$$x_{jkt} \leq z_{jk,\lfloor t/m \rfloor}, j \in V, k \in R_j, t = 0, \ldots, T-1 \tag{6.6}$$

$$\sum_{j \in V} \sum_{k \in R_j} \sum_{t \in T_d} a_{ijk} x_{jkt} \leq 1, i \in N, d \in D \tag{6.7}$$

$$\underline{p}_r \leq \sum_{k \in R_j} \sum_{h=0}^{h_r} a_{ijk} z_{jk,(\lfloor t/m \rfloor + h) \mod |D|} \leq \overline{p}_r, i \in N_r, t = 0, \ldots, T-1, r \in F \tag{6.8}$$

$$y_j \in \{0,1\}, j \in V \tag{6.9}$$

$$x_{jkt} \in \{0,1\}, j \in V, k \in R_j, t \in T. \quad (6.10)$$

$$z_{jkd} \in \{0,1\}, j \in V, k \in R_j, d \in D. \quad (6.11)$$

The objective function (6.1) states that total vessels charter and fuel cost must be minimized. The charter cost is much higher that the fuel cost associated with sailing and servicing at an installation, the primary objective is to define the most cost effective fleet composition. Constraints (6.2) ensure that each installation receives the required number of visits during the planning horizon. Constraints (6.3) ensure that total duration of voyages sailed by a vessel does not exceeds the maximum number of days a vessel is available. Constraints (6.4) state that the number of vessels serviced at the supply base on day d should not exceed its maximum. Constraints (6.5) ensure that a vessel does not start on a new voyage before it returned to the supply base from the previous. Constraints (6.6) a linking constraints stating that if a vessel j starts a voyage k at time interval number t then there must be departure of this vessel on this voyage on day t/m. Constraints (6.7) insure maximum one departure to an installation during the day. Constraints (6.8) ensure even spread of departures to each installation during the planning horizon. Finally constraints (6.9), (6.10) and (6.11) impose binary requirements on the variables.

6.2 Large neighbourhood search algorithm with flexible departures

In this section, we introduce a detailed description of the LNS with flexible departures. Our idea is based on the algorithm presented by Shyshou et al. (2012). The algorithm is implemented using `C#` programming language and `.NET` framework 4.5. The Figure 6.1 provide a conceptual flowchart of the algorithm where new and modified procedures are highlighted with red.

6.2.1 Heuristics summary

The algorithm is applied for a given number of restarts, pre-defined by a user. At each restart initial feasible solution is randomly generated (a set of voyages assigned

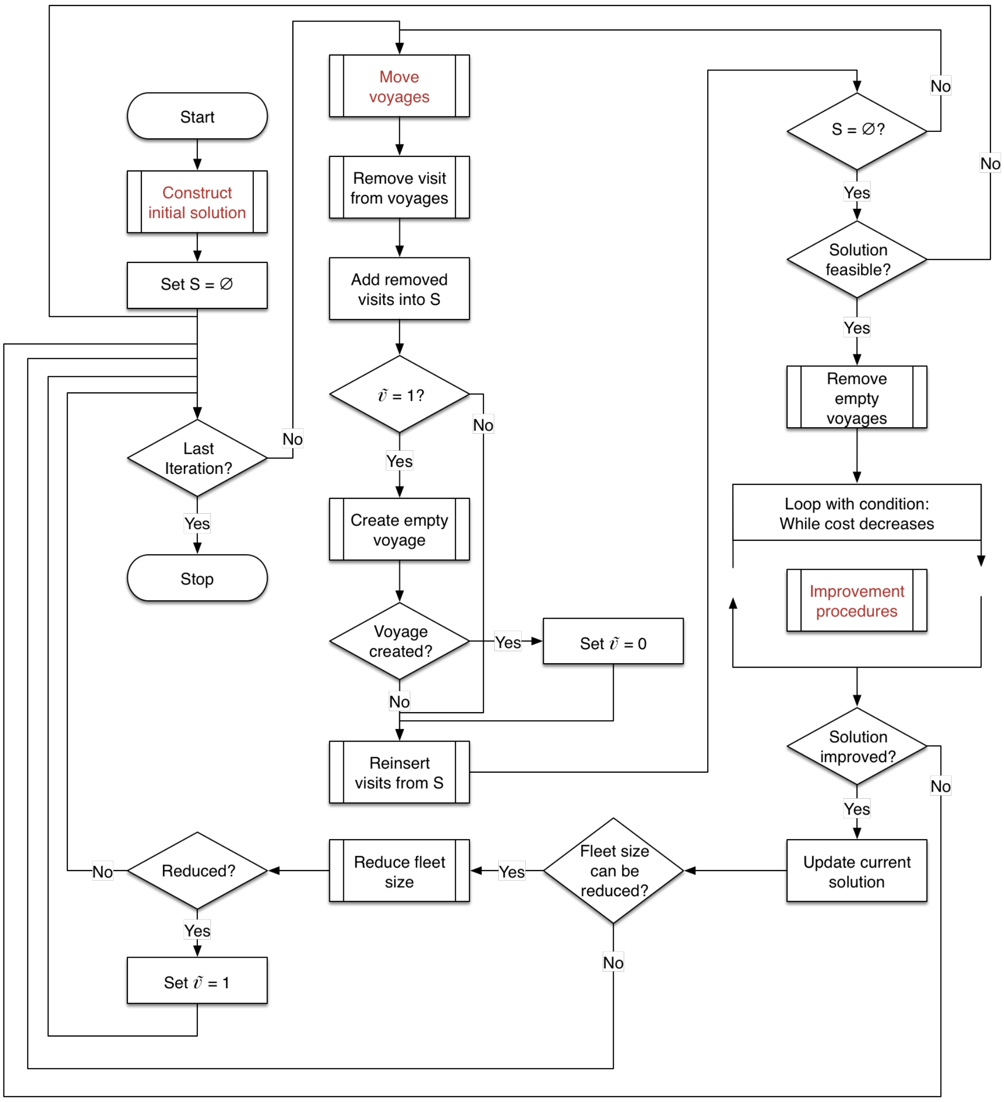


Figure 6.1: A flowchart of the LNS algorithm for the PSVPP with flexible departures

to the PSV for discrete points in time during days of planning horizon) and a number of LNS iterations is performed.

The number of iterations is pre-defined by user as well. At each iteration, the neighbourhood $N(z)$ of a solution z is defined. It means that the area of achievable solutions from solution z is found. In order to find a better alternative of the current solution, a transition from z to neighbourhood solution z' is performed. Such transition is called a move. It is assured by three procedures: *Move Voyages*, *Remove Visits* and *Insert Visits*. The LNS iteration begins with a *Move Voyages*, it takes a fixed number of vessel and for each vessel it remove all its routes, storing them in the pool of uninserted voyages Θ. All stored voyages are evaluated with respect to possible departure times throughout a planning horizon and the idea of packing voyages more tightly to the beginning of the planing horizon. When the best evaluations are identified for all stored voyages and $\Theta = \varnothing$, they are inserted back to the schedule maintaining solution feasibility. Than for the *Remove Visits* and *Insert Visits* procedures, it takes a fixed number of voyages and removes from each randomly generated number of visits and putting them into the pool of uninserted visits S and then reinserting them back into the best position (sequence of visit) of the same or other voyages using a regret criterion. When all removed visits were relocated and $S = \varnothing$, the algorithm performs a set of improvement procedures. If none of constraints were violated at the end it tries to reduce the number of vessels by reassigning voyages from one of the vessel to the other vessels. This procedure may eliminate some voyages, so the lower bound on the number of voyages (calculated at the beginning of the algorithm) is not satisfied. Only in such cases, a new empty voyage will be created before *Insert Visits* procedure takes place on the next iteration. If resulting solution is cheaper than an old one, it becomes a current solution and a new iteration begins. At the end, the algorithm returns the best know solution z^*. All improvement procedures are applied until the cost of current solution z decreases, at the end of each procedure solution feasibility is maintained, such that all constraints should hold (every installation gets its number of required visits, departures are spread uniformly within a week, the capacities of the base and vessel are not violated, the duration of voyages lies in acceptable limits, a vessel is returned and is available at onshore base). Starting from most important, algorithm tries to reduce the number of voyages, and perform a reassignment of voyage to vessel maintaining solution feasibility. Than it comes to the reduction of the total voyage duration and assign voyages to a smaller number of vessels. After that, it tries to relocate visits between voyages to minimize total sailing and service costs. And finally, algorithm check if there are any further improvements in departure times of all voyages after the application of previous improvement procedures and perform a reassignment of departure times for some voyages. The heuristic is outlined in Algorithm 6.2.1, where s is the sum of all voyage durations in days, m – the total number of vessels used, $f_{\min}$ – the minimum number of days one of the used vessels is available.

Algorithm 6.2.1 Main Algorithm

set the cost of the best known solution $c^* = \infty$;
for ρ restarts **do**
construct an **initial solution** z^0 (Procedure 6.2.2);
set $z = z^0$; set $c(z)$ = cost of solution z; set $S = \varnothing$; set $\vartheta = 0$;
for n iterations **do**
if $S = \varnothing$ **then**
remove voyages of some vessels in z (Proc. 6.2.3) and store in Θ;
while there exist feasible insertions of voyages from Θ and $\Theta \neq \varnothing$ **do**
evaluate feasible departure times insertions for each voyage in Θ;
insert voyages back into z (Procedure 6.2.4) and update Θ;
end while
end if
remove visits from some voyages in z (Procedure 6.2.5) and store in S;
if $\vartheta = 1$ **then**
create empty voyages in z (Procedure 6.2.6) and set $\vartheta = 0$;
end if
while there exist feasible insertions of visits from S and $S \neq \varnothing$ **do**
insert visits into voyages in z (Procedure 6.2.7) and update S;
end while
if $S = \varnothing$ **then**
remove empty routes in z;
if z is feasible **then**
while $c(z)$ decreases **do**
reduce the number of voyages in z (Procedure 6.2.10);
reassign voyages to vessel in z (Procedure 6.2.11);
reduce total voyage duration in days in z (Procedure 6.2.12);
reassign voyages to vessel in z (Procedure 6.2.11);
relocate visits between voyages in z (Procedure 6.2.13);
reassign voyages to vessel in z (Procedure 6.2.11);
reassign voyages departure times during a day in z (Procedure 6.2.14);
reassign voyages to vessel in z (Procedure 6.2.11);
end while
if $c(z) < c^*$ **then**
set $z^* = z$; set $c^* = c(z^*)$;
end if
if $\lceil s/f_{\min} \rceil < m$ **then**
reduce the number of vessels in z (Procedure 6.2.8);
If the resulting number of voyages in z does not exceed the minimal number, set $\vartheta = 1$;
end if
end if
end if
end for
end for
return z^*;

6.2.2 Construction of initial solution

Initial solution consists of a collection of feasible voyages starting on a particular day of planning horizon and randomly chosen departure time, which are assigned to some PSV. At each restart initial feasible solution is generated with a random and feasible pattern of departure days, based on the supply base capacity and required number of visits, as well as randomly chosen departure time for each day. All departure patterns are pre-generated ensuring the uniform spread of departures to each installation and required visit frequency, departure times is user-defined option, specified at the beginning of the heuristic. Hence, for each departure day, we construct voyages from a set of installations, assigning randomly chosen departure time and a suitable supply vessel. The order sequence of visits in each voyage is optimized in order to reduce voyage duration. If procedure cannot to find a feasible solution within allowed number of attempts, the fleet size is increased and and procedure is restarted. This is summarized in Procedure 6.2.2.

6.2.3 Move voyages

Move voyages procedure actually consists of two consecutive procedures of removing voyages from the solution z into the pool of uninserted voyages Θ. Evaluating feasible departure times insertions for each voyage in Θ. And re-inserting voyages back into z. The following sections describe this process in more details.

Removing voyages

This procedure make use of user-defined option on number of vessel to be selected. It selects such vessels to be chosen for voyage removal, that the total idle time for weekly routes is minimized. Than, all the voyages of selected vessels removed and stored in the pool of uninserted voyages Θ. This is described in Proc. 6.2.3.

Inserting voyages

This procedure aims to insert voyages back into z from the pool of uninserted voyages in Θ maintaining the solution feasibility. First it evaluates possible combinations of departure times for each uninserted voyage for each vessel, so that the

Procedure 6.2.2 Initial Solution

1: **Input:** all instance data, maximum allowable number of attempts $a_{\max}$
2: set $a = 0$;
3: **repeat**
4: set list of voyages $K = \varnothing$; set $a = a + 1$; set infeasibility flag $f = 1$;
5: **for** each installation $i \in I$ **do**
6: randomly generate a feasible pattern of departure days D_{ti}; $\triangleright$ $D_{ti} = 1$, if installation i is visited from departure day t, 0 otherwise;
7: **end for**
8: calculate maximum number of installation to be visited departuring on any day; set $d_{\max}$;
9: **if** there is such $t \in T$ that $\sum_{i \in I} D_{ti} > d_{\max}$ **then**
10: set $f = 0$;
11: **end if**
12: **if** $f = 0$ **then**
13: **for all** departure days $t \in T$ **do**
14: set λ_t = number of visits per departure day t;
15: set $\lambda_{\max}$ = maximum number of visits allowed for a voyage;
16: **if** $\lambda_t < \lambda_{\max}$ **then**
17: construct a single voyage for all visits to be departing on day t;
18: assign randomly chosen departure time during a day from list of available times; put voyages into K;
19: **else**
20: evenly distribute visits between $\lceil \lambda_t / \lambda_{\max} \rceil$ voyages;
21: assign randomly chosen departure time during a day from list of available times; put voyages into K;
22: **end if**
23: **for all** voyages $k \in K$ **do**
24: call **intra-voyage optimization** (Procedure 6.2.9) to reorder visits and optimize voyage duration;
25: **end for**
26: **end for**
27: **end if**
28: **if** $f = 1$ **then**
29: **for all** voyage $k \in K$ on each day $t \in T$ **do**
30: assign first available vessel with sufficient capacity to k according to a first-fit decreasing rule E.G. Coffman et al. (1984);
31: **if** no available vessel **then**
32: set $f = 0$; break;
33: **end if**
34: **end for**
35: **end if**
36: **until** $f = 0$ or $a < a_{\max}$
37: **if** $f = 0$ or $a > a_{\max}$ **then**
38: increase fleet size and go to line 2;
39: **end if**
40: **return** feasible solution z^0;

Procedure 6.2.3 Remove Voyages

Input: current solution z, pool of uninserted voyages Θ
denote by M the number of vessels to be considered for voyages removal (user-defined);
denote by t the number of chosen vessels;
denote by A the set of all used vessels;
denote by W the set of chosen vessels;
set $t = 0$;
while $t < M$ **do**
    find next vessel in A with least total idle time, but more than 24 hours;
    add vessel to set W
    set $t = t + 1$;
end while
for all vessels $w \in W$ **do**
    denote by R_w the number of voyages of vessel w;
    while $R_w \neq \varnothing$ **do**
        reversibly remove all voyages of w, starting from the last position;
        store voyages to vessels w assignment information;
        insert all removed voyages in Θ;
    end while
end for
return z,Θ;

initial sequence of a vessel voyages are not necessarily respected. For each evaluation any changes of voyages duration, slack or departure times is retained. Than all evaluations are sorted and the best evaluation is identified. If solution is maintained feasible, than the voyages of evaluation are re-inserted back into z. Otherwise, next best evaluation is checked against constraints. Algorithm repeats until there are no possible improvement in departure times yielding objective function improvement. This is described in Procedure 6.2.4.

6.2.4 Removing visits from voyages

Procedure relies on user pre-defined number of voyages to be selected. Than is selects random voyages up to the defined number, from which a random number of visits are removed, from 1 up to all visits. Removed visits are placed into the pool of uninserted visits S. This is described in Procedure 6.2.5.

Procedure 6.2.4 Insert Voyages

Input: current solution z, pool of uninserted voyages in Θ;
denote by W the set of vessels with removed voyages;
denote by E the set of voyages move evaluation;
repeat
    for all vessel $w \in W$ **do**
        for all voyages $k \in \Theta$ of vessel w **do**
            for each day $t \in T$ **do**
                if voyage k can be feasibly start on day t **then**
                    Change voyages departure times during a day (Procedure 6.2.15);
                    store the corresponding option o for voyage k in Ω;
                end if
            end for
        end for
        denote by Ψ feasible combinations of voyages options o for vessel w;
        for feasible option combinations $\psi \in \Psi$ for vessel w **do**
            perform evaluation of option combination ψ;
            store the corresponding changes of day, duration, slack and departure time changes in evaluation $\epsilon \in E$;
        end for
    end for
    if there are several feasible evaluations $\epsilon \in E$ for each vessel w **then**
        to prioritize maximization of vessel idle period and pack voyages closer to the beginning of the week, sort evaluations ϵ by the largest values of the lexicographical ordering: 1. number of days decrease of voyages; 2. time duration decrease of voyages; 3. smallest start time changes compared to the beginning of the week;
    end if
    for sorted evaluations $\epsilon \in E$ for each vessel w **do**
        tentatively insert voyages of evaluation ϵ into z;
        if departure spread constraint is not violated
          and objective function has decreased **then**
            define ϵ' as the evaluation yielding the smallest objective function;
        end if
    end for
    if $\epsilon' \neq \varnothing$ **then**
        insert voyages of evaluation ϵ' into z;
    end if
until there are no possible departure time change improving the objective function;
return z, Θ;

Procedure 6.2.5 Remove Visits

Input: current solution z, pool of uninserted visits S
randomly choose a subset Θ of voyages;
for all voyages $k \in \Theta$ **do**
  denote by V_k the number of visits belonging to voyage k;
  randomly generate the number of visits $\mu \in [1, V_k - 1]$ to remove from voyage k;
  denote by p the number of removed visits
  randomly select starting position $i \in [1, V_k]$;
  while $p < \mu$ **do**
    remove μ consecutive visits starting with the installation in the i^{th} position of k;
    insert all visits in S;
    if $i = V_{\max}$ **then**
      $i = 0$
    end if
    set $p = p + 1$;
  end while
  call **intra-voyage optimization** (Procedure 6.2.9) for k;
end for
return z,S;

6.2.5 Creating empty voyages

This procedure is applied only if there is a vessel, which was marked as "not used" by Procedure 6.2.8 and if the number of voyages is not larger than the lower bound on the number of voyages in the solution (calculated at the beginning of the algorithm). Than procedure creates as many empty voyages as it is necessary to remove possible solution infeasibility. Empty voyages are created in time slots where some vessels have idle time. This is described in Procedure 6.2.6.

Procedure 6.2.6 Create Empty Voyage (Shyshou et al. 2012)

Input: current solution z
denote by $MinVoyage$ = minimum voyage duration;
for every vessel j used in z **do**
  for every time slot δ **do**
    if $\delta \geq MinVoyage$ **and** j is available during δ
      and previous voyage performed by $j \geq MinVoyage$ **then**
      create an empty voyage and insert it in z;
    end if
  end for
end for
return z

6.2.6 Inserting visits to voyages

The purpose of this procedure is to re-insert visits stored in the pool of uninserted visits S to existing voyages of solution z. It first evaluates possible insertions for each visits of pool S into every voyage. Than it defines all best insertion for every visit v and performs corresponding insertions. Such procedures repeats until the pool of uninserted visits is empty or it is not possible to feasible insertion all visits. This is summarized in Procedure 6.2.7.

Procedure 6.2.7 Insert Visits

Input: current solution z, pool of uninserted visits S;
denote by f the indicator of feasible insertion;
denote by K a set of existing voyages;
denote by E a set of visits evaluations;
while $f = 1$ and $S \neq \varnothing$ **do**
    for all visits v in S **do**
        for all voyages k in K **do**
            if visits v can feasibly be inserted into voyage k **then**
                perform tentative insertion v at the beginning of k;
                call **intra-voyage optimization** (Procedure 6.2.9) for voyage k;
                evaluate insertion cost of visit v into voyage k and voyage k duration;
                store evaluation ϵ in E;
            end if
        end for
        if there are several evaluations $\epsilon \in E$ **then**
            calculate the regret values for each evaluations ϵ;
        end if
    end for
    if there are visits in S that only one possible insertion **then**
        define ϵ' as evaluation yielding the smallest values of the lexicographic ordering: 1 duration increase of the destination voyage; 2. objective function increase;
    else
        define ϵ' as evaluation with the smallest duration increase of the destination voyage and largest regret value;
    end if
    perform suggested insertion of evaluation ϵ' visits v into voyage k;
    call **intra-voyage optimization** (Procedure 6.2.9) for voyage k;
    set $S = S \backslash v$;
end while
return z,S;

6.2.7 Reducing number of vessels

The procedure tries to get rid of some supply vessel via reassigning all its voyages to other vessels. If this is impossible, the procedure identifies a vessel used a least number of days after partial reassignment and removes all visits of this vessel into the pool S, marking the vessel as "not used". This is described in the Procedure 6.2.8.

Procedure 6.2.8 Reducing Number of Vessels

```
Input: current solution z, pool of uninserted visits S;
denote by A the set of all used vessels;
for all vessel a ∈ A do
    perform evaluation of partial reassignments of voyages of vessel a to
    other vessels in A;
end for
denote by B the best evaluation of partial reassignments of voyages
perform the partial reassignment of the best evaluation B;
remove all visits from all voyages of vessel a put them into S;
set the status of vessel a to "not used";
return z,S;
```

6.2.8 Improvement procedures

Intra voyage optimization

This procedure represents a local improvement heuristic, it aims to optimize visit sequence of a voyage, so that the duration of the voyage is reduced. During the optimization process, it removes each visit of voyage and finds the first position to insert it back in such a way that voyage duration is reduced. The procedure is repeated while there are some voyage duration reduction. This is summarized in Procedure 6.2.9.

Reducing the number of voyages

Procedure 6.2.10 tries to relocate visits from the different voyages into other voyages until source voyage is empty and could be removed. It is applied only if the number of voyages in solution z is higher than the lower bound of possible number of voyages $\underline{n}$ (calculated at the beginning of the algorithm).

Procedure 6.2.9 Intra Voyage Optimization

Input: voyage k
while there are relocations reducing the voyage duration **do**
  for all visits $v \in V$ in voyage k **do**
    remove visit v;
    find first position p to insert v so that duration of k reduces;
    if position found **then**
      insert v in position p;
      break;
    end if
  end for
end while
return k;

Procedure 6.2.10 Reducing the Number of Voyages (Shyshou et al. 2012)

Input: current solution z
$\overline{z} = z$;
for every voyage k in $\overline{z}$ **do**
  if number of voyages in $\overline{z}$ is equal to $\underline{n}$ **then**
    return z;
  end if
  $Counter = 0$;
  for every visit v in voyage k **do**
    remove one visit from the voyage;
    try to insert removed visit to another voyage;
    if $\overline{z}$ is feasible **then**
      $Counter = Counter + 1$
    end if
  end for
  if $Counter$ is equal to number of visits in voyage k **then**
    $z = \overline{z}$;
  else
    $\overline{z} = z$;
  end if
end for
return z;

Reassigning voyages to vessels

Procedure 6.2.11 aims to reduce the fleet size by keeping voyages closer to each other and to reduce average fleet capacity by using smaller vessels, so fleet size is tried to be kept as constant.

Procedure 6.2.11 Reassigning Voyages to Vessels (Shyshou et al. 2012)

Input: current solution z;
copy z into $\overline{z}$;
for each vessel j in $\overline{z}$ **do**
    set $Counter = 0$;
    for each voyage k performed by vessel j **do**
        try to reassign k to another used vessel or to a smaller (not necessarily used) vessel;
        if $\overline{z}$ is feasible **then**
            $Counter = Counter + 1$;
        end if
    end for
    if $Counter$ is equal to number of voyages performed by vessel j **then**
        $z = \overline{z}$;
    else
        $\overline{z} = z$;
    end if
end for
return z;

Reducing total duration of voyages

Procedure 6.2.12 attempts to reduce the total duration of the voyages measured in days. Since voyages are measured in integer number of days, some gaps or idle time between voyages of the same vessel may exist. The procedure looks for visits re-assignment to reduce voyage duration time, which is possibly a fraction time of a day, thus this reduction results in a day reduction. Procedure tries to re-insert visits to those voyages, whose duration in days will not be increased.

Relocate visits to other voyages

The Procedure 6.2.13 tries to relocate visits of between voyages in order to reduce the total sailing cost while decreasing the objective function. It evaluates all feasible insertions yielding objective function decrease, and applies best relocations in terms

Procedure 6.2.12 Reducing Total Duration of Voyages (Shyshou et al. 2012)

```
Input: current solution z;
while there are voyages in z whose duration in days can be reduced without
increasing the duration in days of other voyages do
    for every voyage k whose duration is larger than the minimum duration do
        while there are feasible visit relocations
            or until the duration of the voyage is reduced by one day do
            for each visit v belonging k do
                evaluate all feasible relocations to different voyages so that
                the duration of the destination voyage in terms of number of
                days does not increase;
            end for
            to prioritize relocation of "difficult" visits, i.e., visits with fewer re-
            location alternatives, implement the best (the smaller the better)
            relocation in terms of the following lexicographic ordering: 1. num-
            ber of feasible relocations; 2. net voyage duration increase; 3. increase
            in the objective function;
        end while
        if the voyage duration is reduced then
            store the respective change in the objective;
        end if
    end for
    implement the voyage duration reduction yielding the smallest ob-
    jective function increase;
end while
return z;
```

of total cost, this repeats until there are no relocations that improve the objective function.

Procedure 6.2.13 Relocating Visits to Other Voyages (Shyshou et al. 2012)

```
Input: current solution z;
repeat
    for each origin voyage k in z do
        for each visit v belonging to k do
            for each destination voyage l do
                if duration of k after removing v is out of bounds
                    or duration of l after inserting v is out of bounds
                    or vessel capacity of l after inserting v is violated
                    or uniform spread of departures is not maintained for
                     installation associated with v then
                    isFeasible [k,v,l] = 0;
                else
                    isFeasible [k,v,l] = 1;
                    Decrease [k,v,l] = decrease in the objective by moving v
                    from k to l;
                end if
            end for
        end for
    end for
    if Decrease [k*,v*,l*] = max(Decrease [k,v,l]) and Decrease [k*,v*,l*] > 0
        and isFeasible [k*,v*,l*] = 1 then
        implement the relocation of v* from k* to l*;
    end if
until there are no relocations improving the objective function
return z;
```

Reassigning voyages departure times during a day

If some improvement procedures were successful, it may happen that departure at the current time is no longer the best option for the particular day. Procedure 6.2.14 take care of identifying such cases and make appropriate changes in solution in order to minimize the objective function. It is looking for possible departure time adjustments for all voyages of solution z, tentatively changing departure time of voyages. If there are some improvements leading to the minimization of objective function, it applies departure time changes for the routes along with the intra-voyage optimization procedure. These steps are repeated until any further improvement is found.

Procedure 6.2.14 Reassigning Voyages Departure Times During a Day

Input: current solution z;
denote by K a set of existing voyages;
denote by I a set of improved voyages;
while there are possible improvements yielding voyage k duration minimization **do**
    for all voyages in $k \in K$ **do**
        tentatively **change voyages departure times during a day** (Procedure 6.2.15);
        if objective function of solution z has decreased **then**
            store the corresponding change in voyage k into I;
        end if
    end for
    for all improved voyages in I **do**
        apply changes of improved voyages into z;
    end for
end while
return z;

Change departure time during a day

Procedure 6.2.15 changes departure time of voyage k during a specific day d in a way that the duration of the voyage is minimized and voyage is started as early as possible to the begining of the week.

Procedure 6.2.15 Change Departure Time During a Day

Input: voyage k, departure day d, possible departure times H;
for all departure times $h \in H$ **do**
    tentatively change departure time of voyage k to h;
    call **intra-voyage optimization** (Procedure 6.2.9) to reorder visits and optimize voyage duration;
    determine best departure time h' during a day d yielding voyage k duration minimization and starting closer to the begining of the week;
    change departure time of voyage k to h';
end for
return k;

6.3 Large neighbourhood search algorithm with flexible departures and coupled vessels

Planning horizon is typically considered to be one week. However, there might be advantage of vessels utilization allowing PSVs to sail one week routes of 8 or 9 day, and one of 6 or 5 days on the following week. In order to keep voyages the same for both first and second week, algorithm tries to find so-called coupled vessels: the PSVs that can sail alternating routes on consecutive week. The Figure 6.2 provide a conceptual flowchart of the algorithm where new and modified procedures are highlighted with red.

The algorithm of finding coupled vessels solutions is applied at number of restarts and iterations. It takes the best found not-coupled solution from 6.2 as an input. If the amount of used vessels is acceptable to look for a objective function improvement with coupled schedule, the algorithm sends the current schedule through the set of improvements procedures, allowing first and last voyages of the same PSV to overlap each other over the weekends. Algorithm than insures that the number of vessels with overlapped voyages does not exceed theoretically allowed number. Than is looks for vessel to be paired with the one with overlap, so that both vessels may sail voyages of each other on following week. Algorithm than stores the resulting schedule as the best found, insuring the cost benefit of schedule with coupled PSVs and executes the next restart. The final schedule with coupled PSVs will be the best found across all restarts.

In order support schedules with coupled vessels, all improvement procedures were modified to allow the overlap over weekends between first and last voyage of the same vessel. Modifications were made in such a way to support both coupled and not coupled algorithms, so the general logic remains the same.

When we relax the constraints, in this case, allowing the existence of overlapped voyages while looking for the schedule with coupled PSVs, sometimes it may happen that a schedule without coupled PSVs is generated. This can be explained by the fact that violation of constraints may help the LNS heuristic to avoid trapping into the local minimum, If that happens and the object objective function is lower compared to the schedule taken as the basis. Algorithm stores such schedules, so the final schedule would be the best found the among schedule with and without coupled PSV in terms of objective function. We summarize the algorithm in Procedure 6.3.1.

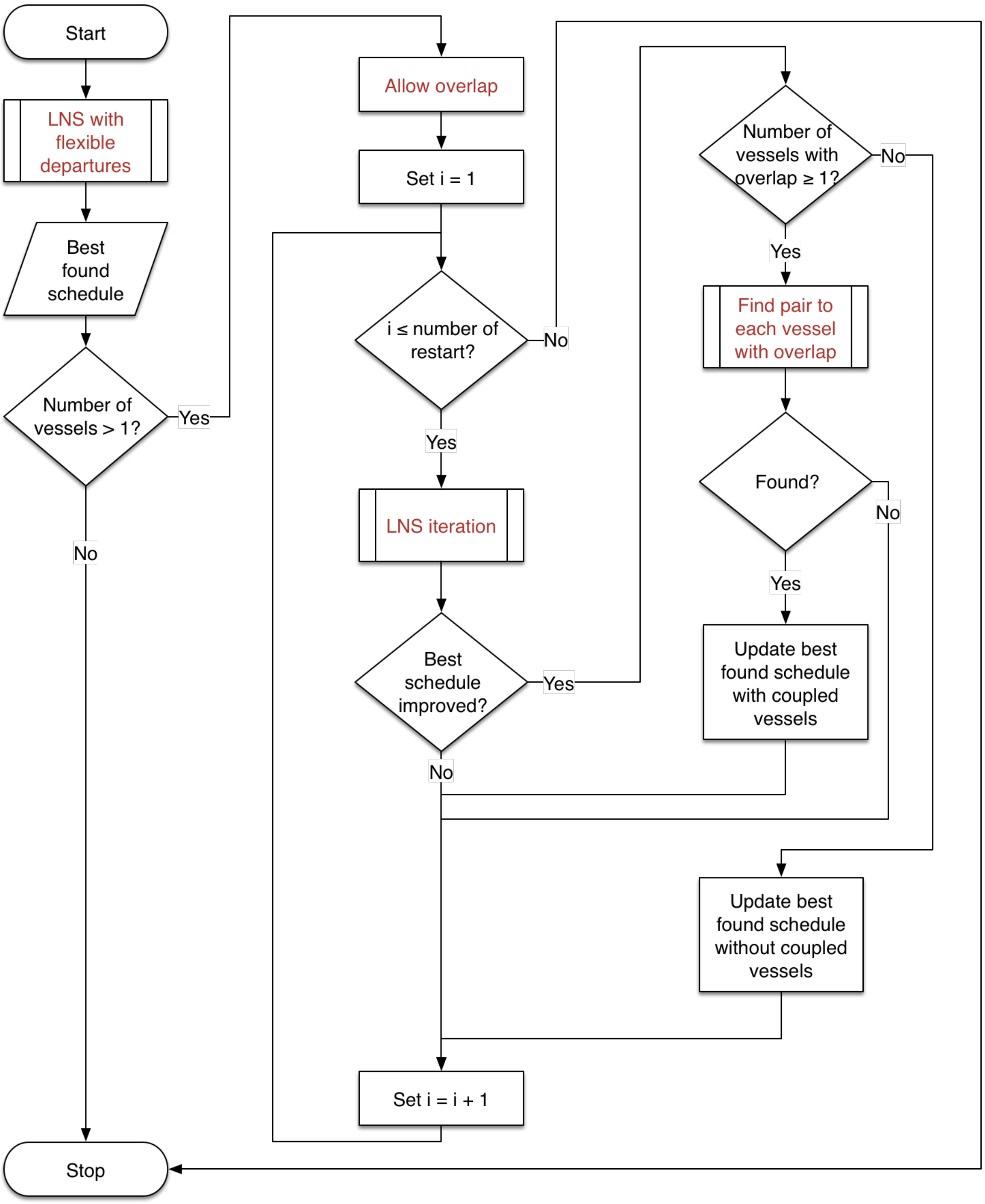


Figure 6.2: A flowchart of the LNS algorithm for the PSVPP with flexible departures and coupled vessels

Algorithm 6.3.1 The LNS with Coupled Vessels

Input: current solution z_{Ncoup};
denote by A the set of all used vessels;
denote by $V^o_{\max}$ the maximum possible number of vessels with overlapped voyages;
if number of vessels $j \in A > 1$ **then**
    set the cost of the current solution $c^*_{Ncoup} = c(z_{Ncoup})$;
    set the cost of the best known solution with coupled vessels $c^*_{coup} = \infty$;
    for ρ restarts **do**
        copy z_{Ncoup} into $\overline{z}$;
        allow the overlap over weekends between first and last voyage of the same vessel;
        apply n **LNS iterations** for $\overline{z}$ (Procedure 6.2.1, lines 5-42);
        if $c(\overline{z}) < c(z_{Ncoup})$ **then**
            if there are vessels in $\overline{z}$ which routes overlap over weekend **then**
                set V^o = number of vessels with overlap;
                if $V^o \leq V^o_{\max}$ **then**
                    try to **find vessel pairs** (Procedure 6.3.2);
                    if all pairs for overlapped vessel were found **then**
                        set $z_{coup} = \overline{z}$
                    end if
                end if
            else
                set $z_{Ncoup} = \overline{z}$
            end if
        end if
        if solution with coupled vessels was found **and** $c(z_{coup}) < c^*_{coup}$ **then**
            set $z^*_{coup} = z_{coup}$;
            set $c^*_{coup} = c(z^*_{coup})$;
        else if solution without coupled vessels was found
            and $c(z_{Ncoup}) < c^*_{Ncoup}$ **then**
            set $z^*_{Ncoup} = z_{Ncoup}$;
            set $c^*_{Ncoup} = c(z^*_{Ncoup})$;
        end if
    end for
else
    if the solution with coupled vessels cannot be found, **return** z_{Ncoup}
end if
return best of z^*_{coup} **or** z^*_{Ncoup};

Algorithm 6.3.2 Find Pairs for Overlapped Vessels

Input: current solution $\overline{z}$, vessels $j \in A$;
for all vessels $j \in A$ **do**
    define by R_j the voyages of vessel j
    define by d_j the total demand of vessel j voyages
    if voyages of R_j overlap over weekend **then**
        for all vessels $i \in A \backslash \{j\}$ **do**
            define by R_i the voyages of vessel i
            define by d_i the total demand of vessel i voyages
            if voyages of R_i does not overlap over weekend
                and capacities are not violated **then**
                check if the time of last voyage in $R_j \leq$ start time of first voyage of R_i **and** time of last voyage in $R_i \leq$ start time of first voyage of R_j;
                define a vessels pair;
            end if
        end for
    end if
    if at least one pair vessel for each vessel with overlap is found **then**
        Store all possible pairs in V_{ij};
    end if
end for
return set of possible vessel pairs V_{ij};

7 Computational Experiments

In this section we provide description of the conducted experiments. There are several objectives of the experiments. The first, is the validation of the developed heuristic i.e. assessment of the efficiency of the heuristic solution compared to two-phase method solution. The second is to show how implemented modifications yield the cost of the solution. Modifications are introduced in two steps: implementation of flexible departures and then the concept of coupled vessels. And finally we need to understand which settings in terms of the number of restarts and iterations are preferably applied depending on the trade-off between the cost of the solution and computational time.

7.1 Experiments setup

According to one of the objectives, the developed heuristic is to be able to solve the problems of a large size. Since that computational complexity increases with the problem size, validation of the heuristic i.e. comparison of the results of the two-phase method approach to the results provided by the heuristic is only possible for the small and medium size instances. All tests were conducted using a computer with the following configuration: 2.6GHz Intel Core i5, 16GB of 1600MHz DDR3 memory and Microsoft Windows 7 operating system.

7.1.1 Heuristic assessment approach

The following experiments are conducted with the aim of heuristic validation and assessment of the efficiency of flexible departures for the small and medium size instances:

1. Comparison of the solutions provided by the two-phase method with fixed departure times to the approach with flexible departure times;

- The aim is to assess how flexible departure times perform compared to fixed departure time;

2. Heuristic solutions with flexible departures are compared to solutions provided by the two-phase approach with flexible departures;

   - The aim of the experiment is to assess the efficiency of the developed heuristic i.e. to show how the heuristic solution differs from the solution provided by the two-phase approach on the same instances;

As regards instances of the large size, only heuristic solution is available. Experiments setup for the large size instances are as follows:

3. Heuristic solutions with fixed departure times are compared to solutions with flexible departure times;

   - The aim is to assess the cost difference of heuristic solutions with fixed and flexible departures for large size instances;

4. Heuristic solutions with flexible departure times are compared to solutions with flexible departure times and coupled vessels;

   - The aim is to assess the contribution to the cost reduction of coupled vessels.

7.1.2 Assessing the impact of a number of restarts and iterations on the cost of the solution and computation time

Since the heuristic is performed for a given number of restarts and iterations, it is important to understand how the number of restarts and iterations influences the cost of the solution and computational time. The logic of such experiment is to analyze how the solution is improved with the increase of the number of restarts and iterations and to show the relationships between restarts, iterations, solution cost and computational time.

7.2 Test instances

7.2.1 Test instances generation

Test instances are generated based on the information provided by Statoil ASA. The developed heuristic is used to define a weekly sailing plan for oilfields located on the Norwegian continental shelf and served from Mongstad supply base, including in total 26 installations. Figure 7.1 shows the location of all given fields (highlighted with yellow color), assigned offshore installations (marked with red color circles) and the supply base (marked with red triangle). The company provided the following data:

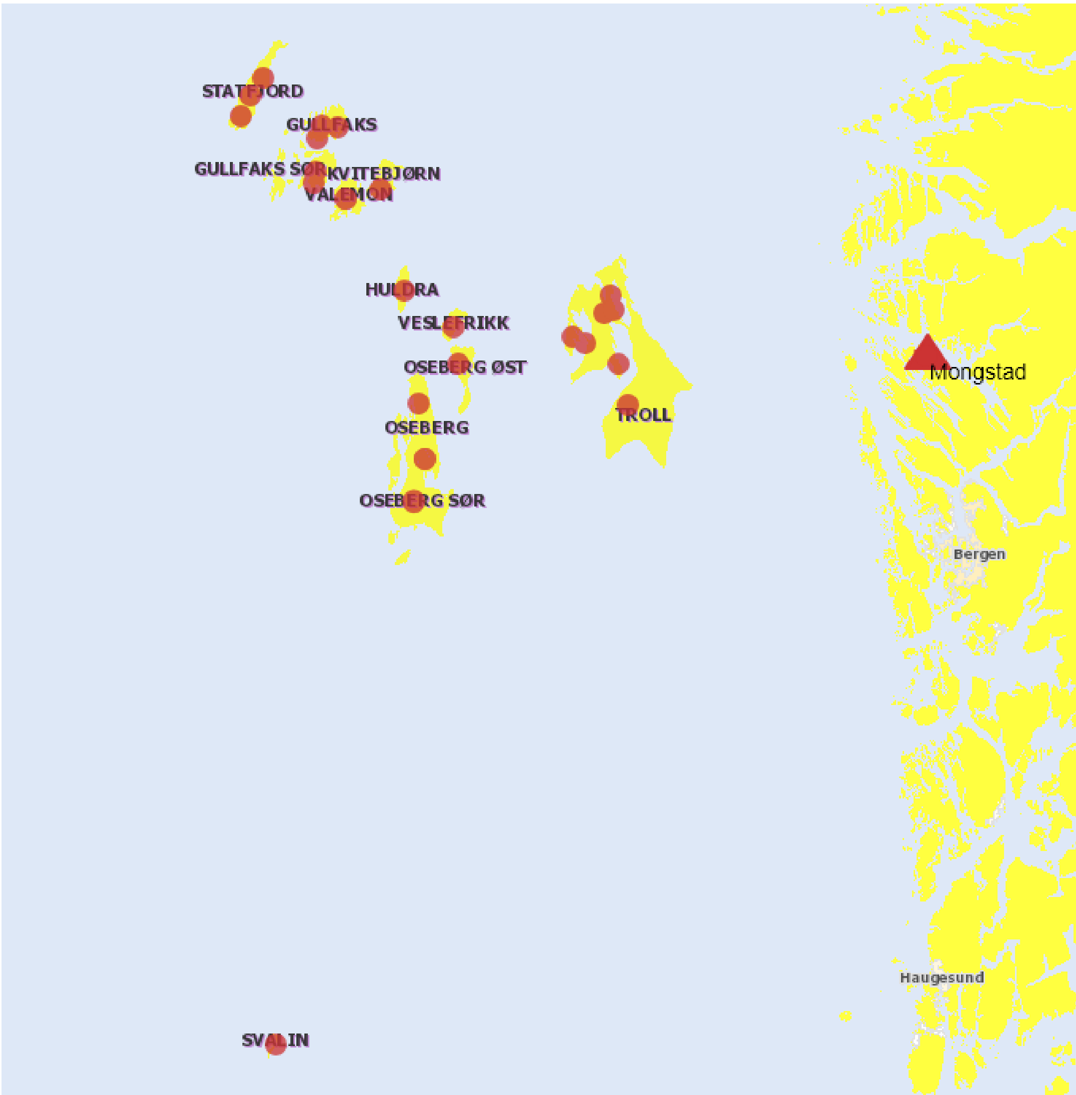


Figure 7.1: Location of offshore installations and supply base

- Offshore installations:
  - Coordinates;
  - Opening and closing hours (if any);
  - Weekly demand (considered to be pretty low);
  - Required number of visits per week;
  - Lay time i.e. the average service time for a vessel at each installation;
  - Departure spread patterns for each visit frequency;
- Supply base:
  - Coordinates;
  - Capacity i.e. the maximum number of vessels that may be served during the working day;
  - Vessel charter cost;
  - Departure time from the supply base (for problems with fixed and flexible departures);
- Supply vessels:
  - Speed;
  - Deck capacity;
  - Fuel consumption rate per hour (sailing and servicing);
  - Fuel cost.

For validation and efficiency assessment of the developed heuristic, the following test instances were generated in terms of the number of installations and number of supply vessels:

- For small and medium size:
  - 27 instances with the number of installations from 3 to 11 (3 different instance setups for each number of installation);
  - Number of available vessels is 3;

- For large size:
  - 45 instances with the number of installations from 12 to 26 (3 different instance setups for each number of installation);
  - Number of available vessels is 7.

As regards the experiments for the analysis of the influence of the number of restarts and iterations on the total cost and computational time we take the instance with 26 installations.

Variations on the number of restarts are: 1, 3, 5, 10, 15 and from 20 to 100 with an interval of 10. For each variation of the restarts above indicated, we run the heuristic for the following variations on the number of iterations: from 10 to 100 with an interval of 10 and from 100 to 1200 with an interval of 100.

7.2.2 Input data description

In this section we illustrate an example of generated test instance on the input data. The data on opening and closing times, demands, required visit frequency per week, lay time (service time) and coordinates for all installations are provided on the Figure 7.2. Each voyage is limited to have minimum one and maximum seven installations. The load factor indicates the multiplier on the demand value, so the capacity of the vessel could be controlled more accurately. The acceptance time represents the tolerance time by which some rules might be violated, i.e in case of voyage overlap, the difference between the finish time of current voyage and the begining time of the next voyage and should not exceed the acceptance time value. Minimal slack represents the minimal slack between voyages. For the problem with fixed departure times, the supply base working hours are 8:00-16:00. Departure time of a vessel from the base is 16.00. For the problem with flexible departures, the supply base working hours are 0:00-24:00 and available departure times are 8:00 and 16:00. Minimum and maximum voyage duration is considered to be 18 and 72 hours respectively. The maximum number of departures per one day is assumed to be 3.

Vessels capacities, speed, fuel costs (NOK/ton), sailing fuel consumption (ton/h), fuel consumption at the base (ton/h), fuel consumption at an installa-

```
MinInst         1
MaxInst         7
LoadFactor      1
AcceptanceTime  0.01
MinSlack        0
OffshorePoint   60.84416667 4.574444444

Node  Open  Close Demand Frequency  LayTime LatDec      LonDec
FMO   0     24    0      0          8.00    60.79446667 5.06300000
GFA   0     24    10     4          3.5     61.17000000 2.18000000
GFB   7     19    10     4          2.5     61.20000000 2.20000000
GFC   0     24    10     4          3.5     61.20000000 2.27000000
STA   0     24    10     3          3.5     61.25000000 1.85000000
STB   0     24    10     3          3.5     61.20000000 1.82000000
STC   0     24    10     3          3.5     61.29000000 1.90000000
DSA   0     24    10     3          3.0     61.09650000 2.18911000
SOD   0     24    10     3          3.0     61.07000000 2.19000000
KVB   0     24    10     4          2.5     61.07000000 2.50000000
VAL   7     19    10     3          2.5     61.04000000 2.34000000
WEL   0     24    10     3          3.0     61.04000000 2.34000000
OSE   7     19    10     2          4.0     60.48000000 2.82000000
OSB   0     24    10     6          2.0     60.48000000 2.82000000
OSC   0     24    10     5          3.0     60.60000000 2.77000000
OSS   0     24    10     4          3.0     60.38000000 2.79000000
OSO   7     19    10     3          3.0     60.70000000 2.93000000
DEL   0     24    10     3          4.0     59.14000000 2.41000000
HUL   7     19    10     1          2.0     60.85000000 2.65000000
VFB   0     24    10     3          4.0     60.78000000 2.89000000
TRO   7     19    10     3          3.0     60.64000000 3.72000000
TRB   7     19    10     2          3.0     60.77000000 3.50000000
TRC   7     19    10     2          3.0     60.88000000 3.60000000
COI   0     24    10     5          4.0     60.84000000 3.58000000
CPR   0     24    10     5          4.0     60.73000000 3.66000000
SDO   0     24    10     5          4.0     60.85000000 3.62000000
WVE   0     24    10     5          4.0     60.78000000 3.44000000
```

Figure 7.2: Offshore installations input data example

tion (ton/h) and vessels charter costs ($NOK/week$) are shown on the Figure 7.3. Departure spread patterns for each visit frequency is provided on the Figure 7.4.

Vessel	Capacity	Speed	FCCosts	FCSailing	FCBase	FCInstallation	VesselCost
RemStadt	1000	10	6000	0.5	0.1	0.4	1400000
TBNSpot	1000	10	6000	0.5	0.1	0.4	1400000
FarStar	1000	10	6000	0.5	0.1	0.4	1400000
VikingEnergy	1000	10	6000	0.5	0.1	0.4	1400000
BourbonTampen	1000	10	6000	0.5	0.1	0.4	1400000
FarSymphony	1000	10	6000	0.5	0.1	0.4	1400000
SkandiFlora	1000	10	6000	0.5	0.1	0.4	1400000

Figure 7.3: Vessels input data example

```
1: 1 2 3 4 5 6
2: 1 4   1 5   2 5   2 6   3 6
3: 1 2 5   1 3 5   1 3 6   1 4 5   1 4 6   2 3 6   2 4 6   2 5 6
4: 1 2 4 6   1 3 4 6   1 3 5 6   1 2 4 5   1 2 5 6   2 3 5 6
5: 1 2 3 4 6   1 3 4 5 6   1 2 3 5 6   1 2 4 5 6
6: 1 2 3 4 5 6
```

Figure 7.4: Departure spread patterns example

7.3 Comparative analysis of results

7.3.1 Application of flexible departure times for two-phase method

The Table 7.1 summarizes the results of the experiments conducted for the PSVPP with fixed and flexible departures provided by the two-phase method for small and medium size instances. The table contains information on the gaps between total costs of the fixed and flexible departure times of the two-phase method, as well as computational time and number of vessels used. The name of an instance consists of four digits divided by hyphen: the first is the number of installations, the second is the number of available vessels, the third – the total number of installations visits and the forth is the number of installation with time windows, which are generally closed during night time (from 19:00 till 7:00).

First, we assess the efficiency of flexible departures. As it is seen from the Table 7.1, for instances with 3, 4, 5 and 6 installations, the gap between solution costs of the two-phase method (fixed and flexible) accounted for 40%–50%. Such gap

Table 7.1: Comparison results of the two-phase method with fixed and with flexible departure times

Instance	Gap (%)	CPU (sec)		Number of Vessels	
	Fixed vs. Flexible	Fixed	Flexible	Fixed	Flexible
3-3-10-1	44,98	0	1	2	1
4-3-11-2	44,91	0	1	2	1
5-3-14-2	44,48	0	1	2	1
6-3-16-3	44,06	0	0	2	1
7-3-23-1	0,05	67	87	2	2
8-3-27-3	0,1	78	123	2	2
9-3-31-3	0,08	91	538	2	2
10-3-34-4	0,84	1253	1433	2	2
11-3-37-4	0,04	2154	2591	2	2
Average	**19,95**	**404,78**	**530,44**	**2,00**	**1,56**

can be explained by the reduction of the fleet size. Two-phase method with fixed departures provided solutions with two vessels for four instances while the flexible departure times provided solutions with only one. For instances from 7 up to 11 installations, the two-phase method with flexible departures provided lower costs, but the gap is small, since the number of vessels is the same for both methods, on the other hand costs reduction is caused by lesser fuel consumption. Such low gap could be explained by the major cost contributor of charter cost, e.i. for instance with 11 installations total charter cost is 3,66 times greater than the total fuel consumption costs. Solution schedules with 6 installations for fixed and flexible departure times are depicted on the Figures 7.5 and 7.6. Schedules costs are NOK 3175920 and NOK 1776698, which generates a cost decrease of 44,06%. Both schedules have 16 visits, which are performed within 4 voyages, but the advantage of using flexible departure times reduces one vessel. On the figures, each voyage starts and ends at the supply base, marker by FMO label, all other labels are related to the names of offshore installations. Dark green color stands for servicing or waiting time at the supply base or installations. Light green represents sailing time and finally, beige color denotes the idle time between voyages.

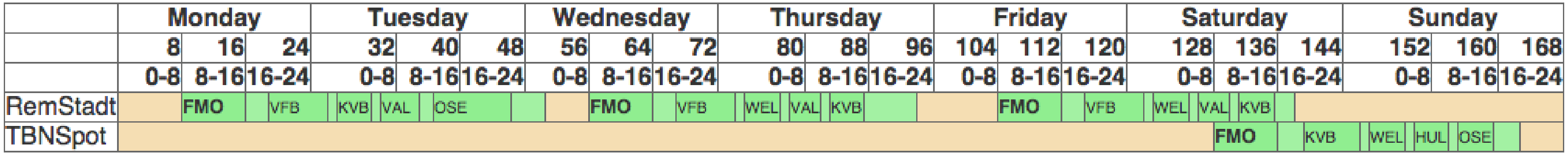


Figure 7.5: Solution schedule with fixed departure times for 6 installations

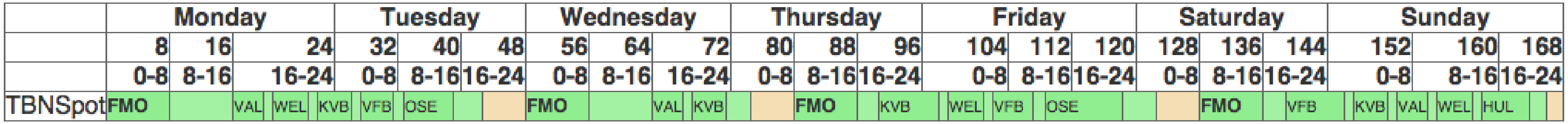


Figure 7.6: Solution schedule with flexible departure times for 6 installations

7.3.2 Validation of the LNS heuristic with flexible departures

For all experiments which includes the LNS heuristic, the following settings of restarts and iterations were used for both LNS with fixed and flexible departure times: for instances with 3 and 4 installations – 20 restarts and 10 iterations. For instance with 5-8 installations – 20 restarts and 20 iterations. For instance with 9, 10 and 11 installations – 20 restarts and 150 iterations. And for all the rest instances number of iterations and restarts are 20 and 300 accordingly.

The Table 7.2 shows the comparison results of the experiments for flexible departures between the two-phase method and the LNS for small and medium size instances. The table contains the information on the gap between the total costs of the two-phase method and the LNS heuristic, computational time and number of used vessels for each approach for each instance size.

Table 7.2: Comparison results of the two-phase method and the LNS with flexible departure times

Instance	**Gap (%)**	**CPU (sec)**		**Number of Vessels**	
	LNS vs. Two-phase	**Two-phase**	**LNS**	**Two-phase**	**LNS**
3-3-10-1	0	1	0	1	1
4-3-11-2	0	1	1	1	1
5-3-14-2	0	1	2	1	1
6-3-16-3	0	0	6	1	1
7-3-23-1	0	87	13	2	2
8-3-27-3	0	123	13	2	2
9-3-31-3	0	538	55	2	2
10-3-34-4	0,06	1433	33	2	2
11-3-37-4	0,07	2591	57	2	2
Average	**0,01**	**530,44**	**0,22**	**1,56**	**1,56**

Now we assess performance of LNS with flexible departures against two-phase method with flexible departures. As we see, for instances with the number of installations from 3 to 9, the gap between two-phase and LNS solutions with flexible departures is zero i.e. the heuristic provides optimal solution. For instances with 10 and 11 installations, the gap is 0.06% and 0.07% respectively that is very insignificant. As regards the computational time, for small instances (3-6 installations) the difference is minor. However, as regards medium size instances, heuristic performs considerably faster than the two-phase approach with a zero or near zero gap. For example, heuristic performs 45 times faster than the two-phase method for the instance with 11 installations (2591 sec against 57 sec).

7.3.3 Application of LNS heuristic with flexible departures on large size instances

The Table 7.3 presents the result of the experiments for the large size instances for LNS heuristics with fixed and flexible departure times. The table depicts gaps between two types of costs (the total costs that includes charter cost and fuel cost and the gap only between fuel costs), computational time and the number of used vessel for each instance. As we see the LNS with flexible departures performs better than the LNS with fixed departure time. The average gap between fuel costs is 1,45%. As regards the total costs, the average gap is 4,12%. Such gap is explained by the fact that LNS with flexible departure managed to save one vessel for the instances with 16, 22 and 23 installations compared to the results provided by the LNS with fixed departure, gaps are 21%, 16% and 15% respectively. Computational time increased, of course, but not significantly. For the instance with 13 installations the gap is only 25 seconds and for the largest instance the gap is 130 seconds, which is a bit more than two minutes.

Solution schedules with 16 installations for fixed and flexible departure times are shown on the Figure 7.7 and 7.8. Schedules costs are NOK 6740586 and NOK 5349913 respectively, which generates a cost decrease of 20,63%. Both schedules have 60 visits, which are performed within 9 voyages by 4 vessels in case of fixed departure algorithm and 3 vessels in case of flexible departure times. Another example on a larger instance of 23 installations is presented on the Figure 7.9 and 7.10. Schedules costs are NOK 8464933 and NOK 7070435 respectively, the cost decrease is 16,47%, 76 visits are performed within 12 voyages. Schedule with flexible departures are able

Table 7.3: Comparison results of the LNS heuristic with fixed and flexible departure times on large instances

Instance	**Gap (%)**		**CPU (sec)**		**Vessels**	
	Fuel cost	**Total cost**	**Fixed**	**Flexible**	**Fixed**	**Flexible**
13-7-48-4	1,69	0,29	193	218	3	3
14-7-51-4	2,9	0,52	163	190	3	3
15-7-54-4	1,46	0,27	182	171	3	3
16-7-60-6	-0,82	20,63	387	191	4	3
17-7-61-7	1,32	0,21	344	264	4	4
18-7-64-7	1,12	0,18	274	337	4	4
19-7-66-8	2,03	0,35	279	308	4	4
20-7-72-8	3,3	0,63	225	295	4	4
21-7-69-7	3,94	0,79	260	291	4	4
22-7-79-8	-4,06	15,79	287	321	5	4
23-7-76-8	-0,38	16,47	251	398	5	4
24-7-81-8	1,15	0,21	384	401	5	5
25-7-88-8	2,48	0,48	394	338	5	5
26-7-98-8	4,2	0,85	504	634	5	5
Average	**1,45**	**4,12**	**294,79**	**311,21**	**4,14**	**3,93**

to show a vessel reduction (4 vessels used), while the schedule with fixed departures lead to a usage of 5 vessels.

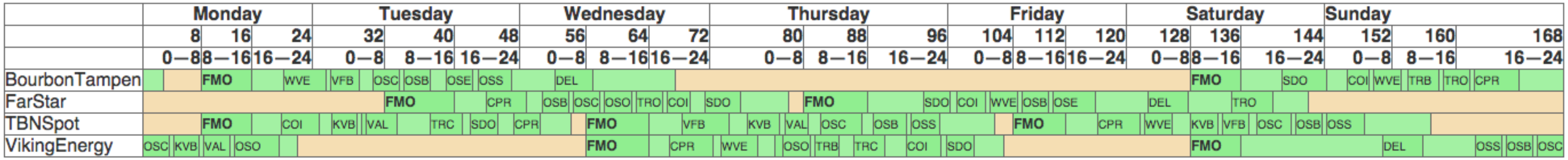


Figure 7.7: Solution schedule with fixed departure times for 16 installations

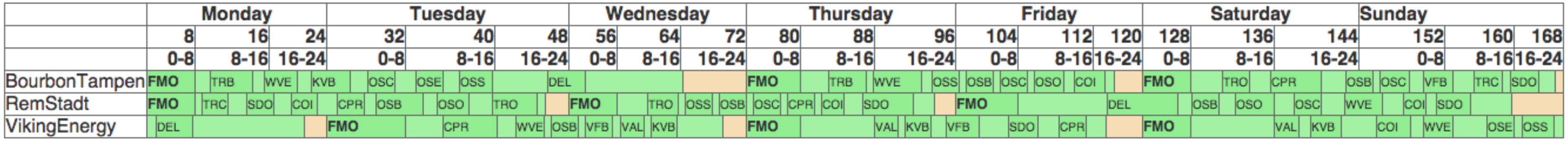


Figure 7.8: Solution schedule with flexible departure times for 16 installations

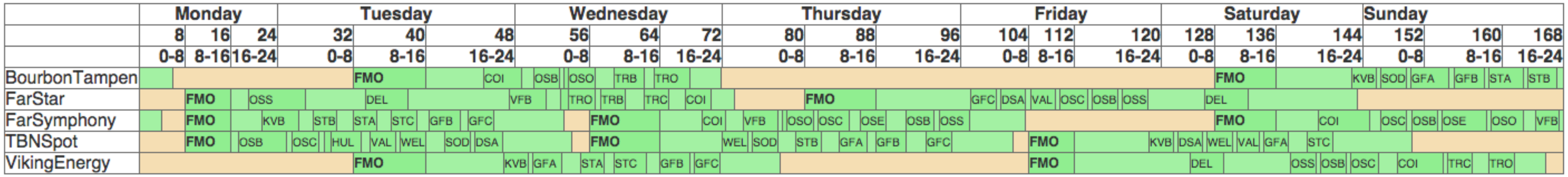


Figure 7.9: Solution schedule with fixed departure times for 23 installations

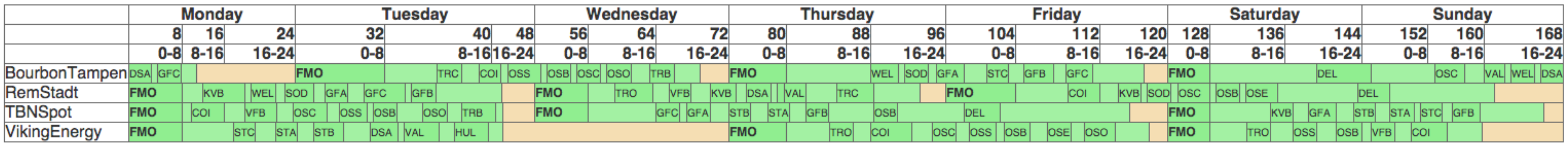


Figure 7.10: Solution schedule with flexible departure times for 23 installations

7.3.4 Application of LNS heuristic with flexible departures and coupled vessels

The Table 7.4 shows the results of experiments conducted for comparison of the LNS with flexible departures and the LNS with flexible departures and coupled vessels. As we see heuristic with flexible departures and coupled vessels managed to find schedules with lower costs compared to heuristic with flexible departures only for the instances with 15, 20, 21, and 24 installations. Implementation of the concept of coupled vessels did not result in the fleet size reduction for all instances. Computational time increased at least twice. Such increase caused by the fact that LNS with flexible departures and coupled vessels uses the output of LNS with flexible departures as initial solution and than a part of the core LNS algorithm is re-applied with an appropriate relaxation allowing PSVs sailing alternating routes on the following week. The average gap between total costs is 0,45% and the gap between fuel costs is a bit higher and accounts for 2,36%.

Table 7.4: Comparison results of the LNS heuristic with flexible departure times and the LNS heuristic with flexible departure times and coupled vessels

Instance	**Gap (%)**		**CPU (sec)**		**Vessels**	
	Fuel cost	**Total cost**	**Flexible**	**Coupled**	**Flexible**	**Coupled**
15-7-54-4	0,56	0,1	171	670	3	3
20-7-72-8	2,36	0,44	295	1087	4	4
21-7-69-7	4,93	0.95	291	986	4	4
24-7-81-8	1,59	0,29	401	864	5	5
Average	**2,36**	**0,45**	**289,50**	**901,75**	**4,00**	**4,00**

As we can see, Figures 7.11 and 7.12 represents the solution schedules for 21 installation with flexible departures time and flexible departures plus the concept of coupled vessels. On the figure, possible pairs of coupled vessels are highlighted with blue color. As we can see, vessel *FarStar* have weekly route that longs longer than one week, but the algorithm was able to find two vessels (*RemStadt* and *BourbonTampen*), which can sail *FarStar's* route on the following week and *FarStars* can sail theirs. Schedules costs for one week are NOK 6943039 and NOK 6876818 respectively. The total cost decrease per week is low (0.95%), since algorithm was not able to reduce the vessels fleet size, but on the other hand the fuel cost decrease for one week is quite significant (4,93%). Computational times are 291 and 986 seconds respectively, time is increased by 239% comparing to the LNS with flexible departures. Schedule contains 69 visits performed within 10 voyages by 4 vessels.

Finally, the Table 7.5 depicts the best results achieved by different LNS algorithms application on the instance of 26 installations. All schedules are shown on the Figures 7.13, 7.14, 7.15 and 7.16. The best schedule for the LNS with fixed departures shows the cost of NOK 8701835, computational time is 8227 seconds. For the LNS with flexible departure times, the total cost is NOK 8683993, which is a 0,21% decrease for a total cost and 1,05% for fuel cost, while the time increase is 7%. The best schedule with flexible departures, which was found during the LNS algorithm with coupled vessels gives a total cost of NOK 8661456, which is the 0,46% decrease for the total cost and 2,37% decrease for the fuel cost. The time increase in this case is 57%, that is 1,5 times more than for the algorithm with fixed departure times. As we see, the best overall result was achieved using the LNS with flexible departures and coupled vessels, it shows the total cost of NOK 8648589 and fuel cost of NOK 1648589, that is a 0,61% and 3,13% cost decrease accordingly. The result obtained is not for the sake of time, which increases by 139%. All the schedules have 98 total visits, performed by 5 vessels, but the LNS with coupled vessels have one voyages less, compared to the other solutions.

Table 7.5: Comparison results of the LNS algorithms for 26 installation

LNS Algorithms	**Fuel Cost**	**Total Cost**	**CPU (sec)**	**Voyages**	**Restarts/ Iterations**
Fixed	1701835	8701835	8227	14	90/800
Flexible	1683993	8683993	8816	14	100/1200
Flexible-Improved	1661456	8661456	12880	14	30/1200
Coupled	1648589	8648589	19643	13	90/800

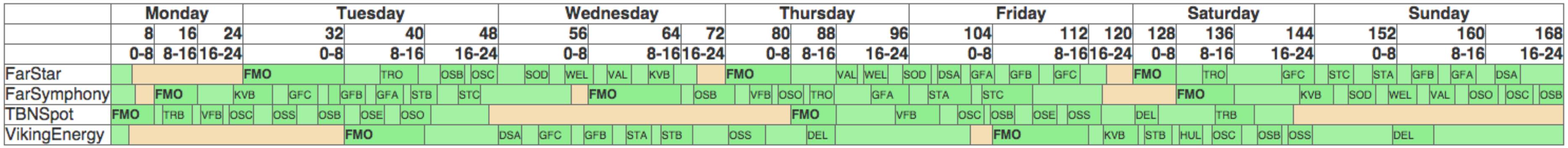
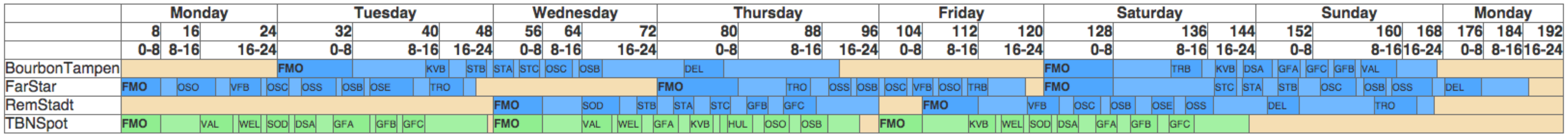


Figure 7.11: Solution schedule with flexible departure times for 21 installations



Vessel	Possible Coupled Vessel
FarStar	RemStadt, BourbonTampen

Figure 7.12: Solution schedule with flexible departures and coupled vessels for 21 installations

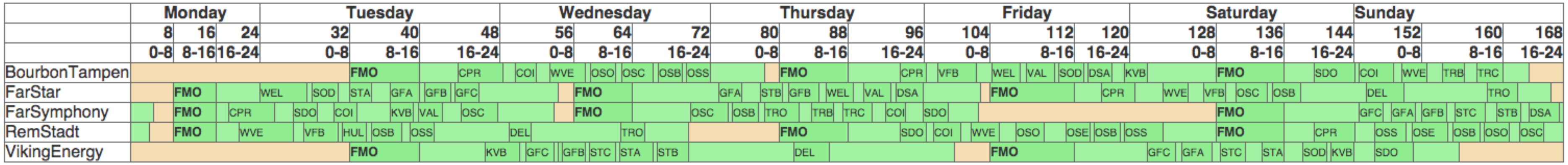


Figure 7.13: Solution schedule with fixed departure times for 26 installations

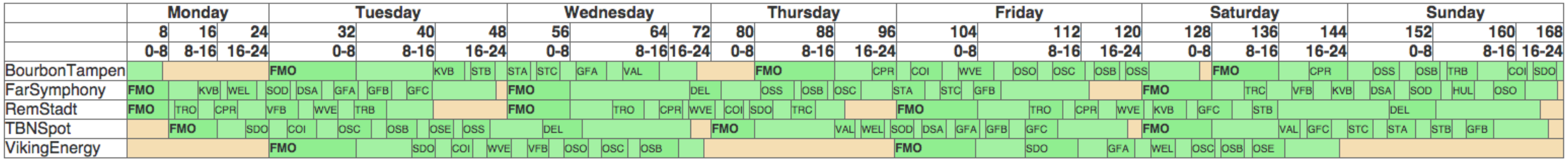


Figure 7.14: Solution schedule with flexible departure times for 26 installations

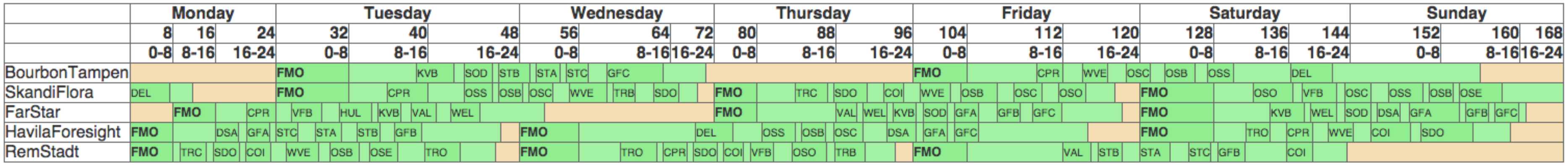
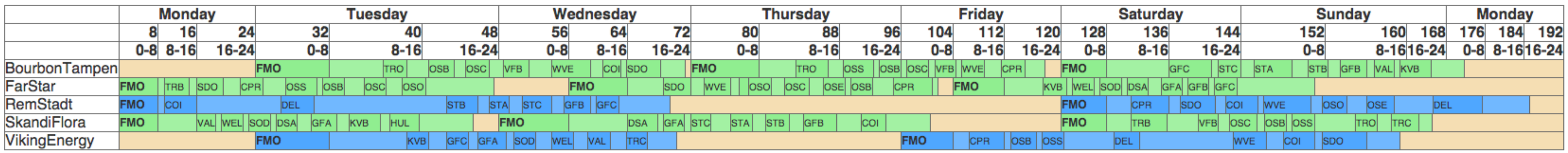


Figure 7.15: Solution schedule with improved flexible departures for 26 installations



Vessel	Possible Coupled Vessel
RemStadt	VikingEnergy

Figure 7.16: Solution schedule with flexible departures and coupled vessels for 26 installations

7.3.5 Evaluation of the cost and computation time with respect to the number of restarts and iterations

There are two user-defined parameters in the LNS heuristic (the number of restarts and number of iterations in each restart), in this section we provide analysis of the results of experiments conducted with the aim of evaluation of the influence of the number of restarts and iterations on the total cost and computational time. The increase of the number of restarts and iterations is supposed to reduce the total cost since the search space for more cost efficient solution increases. However, such cost reduction is obtained, of course, at the cost of computational time increase. Therefore, the problem here is to analyze the influence of restarts and iterations on costs and times. In order to define the appropriate combination of the number of restarts and iterations we have to resolve the trade-off between total cost and computational time. Analysis is provided separately for LNS with flexible departures and LNS with flexible departures and coupled vessels.

Results of the experiments for LNS with flexible departures are summarized in the Appendix A. where for each combination of the number of restarts and iterations provided total cost and computational time. The Table A.1 shows results in terms of the number of vessels used in the solution for each combination of restarts and iterations. We applied color scales for better visualization of the results, so-called heat map. It is simply a colorful representation of the data contained in a matrix, where individual values are highlighted with the color from red to green, showing the increase from the highest value (red) to the lowest (green). We see that in general the increase of the number of restarts and iterations leads to cost reduction and of course computational time increase. From the Table A.1 it is seen that all combinations after 30 restarts and 100 iterations provide results with only five vessels. Vessel charter cost is the major contributor to the total cost and saving of one vessel leads to significant cost reduction. Thus, we may conclude that it is necessary to run the heuristic for at least 30 restarts and 100 iterations to obtain relatively low cost solution. Fluctuations of the total costs for the solutions with five vessels (or for solution with the same number of vessels) are explained by the efficiency of routing and vessel assignment decisions. So, looking at the results strategically we further consider only the results of the solutions with more than 30 restarts and 100 iterations. Combinations with fewer restarts and iterations may provide solutions with the number of vessels from 5 to 7 and do not guarantee relatively low cost. Results for LNS with flexible departures and coupled vessels are summarized in the the Appendix B. Using the same logic for the analysis as for the LNS with flexible departures we may infer that at least 20 restarts and 200 iterations are required to guarantee relatively low cost solution (the solution with minimal number of vessels).

The Figure 7.17 depicts the evolution of the total cost of the solution provided by the LNS with flexible departures depending on the number of restarts and iterations (starting from 20 restarts and 100 iterations). As we see, there is no clear dependence of the total cost with increase of both the number of restarts and iterations. If we have look how the total cost changes for any number of restarts, depending on the change of the number of iterations we see that with the increase of the number of iterations the total cost may decrease or increase. Such unstable behavior might be explained by the random choice presenting in the logic of the some procedures of the algorithm. Nevertheless, if we provide a trend line for each scenario (braking lines of the respective color), we see that all of the lines have a declining trend that means, in general, the increase of the number of iterations for each restart leads to the cost reduction.

If we analyze how the cost changes for a certain number of iterations with the increase of the number of restarts (see Figure 7.18) there is no explicit dependence. With the increase of the number of restarts, the cost may either decrease or increase. If we look at trend lines (braking lines, Figure 7.18) we can see that there is a clear declining trend only for 100, 300, 400 and 1200 iterations. For the rest iterations, there is a slight declining trend and some of them are parallel to X-axis.

If we analyze the results for LNS with flexible departure and coupled vessels (see Figure 7.19 and Figure 7.20) we will see that behavior of the total cost depending on the change of the number of iterations is the same as described above for the LNS with flexible departures, although there is a bit more clear declining trend in the

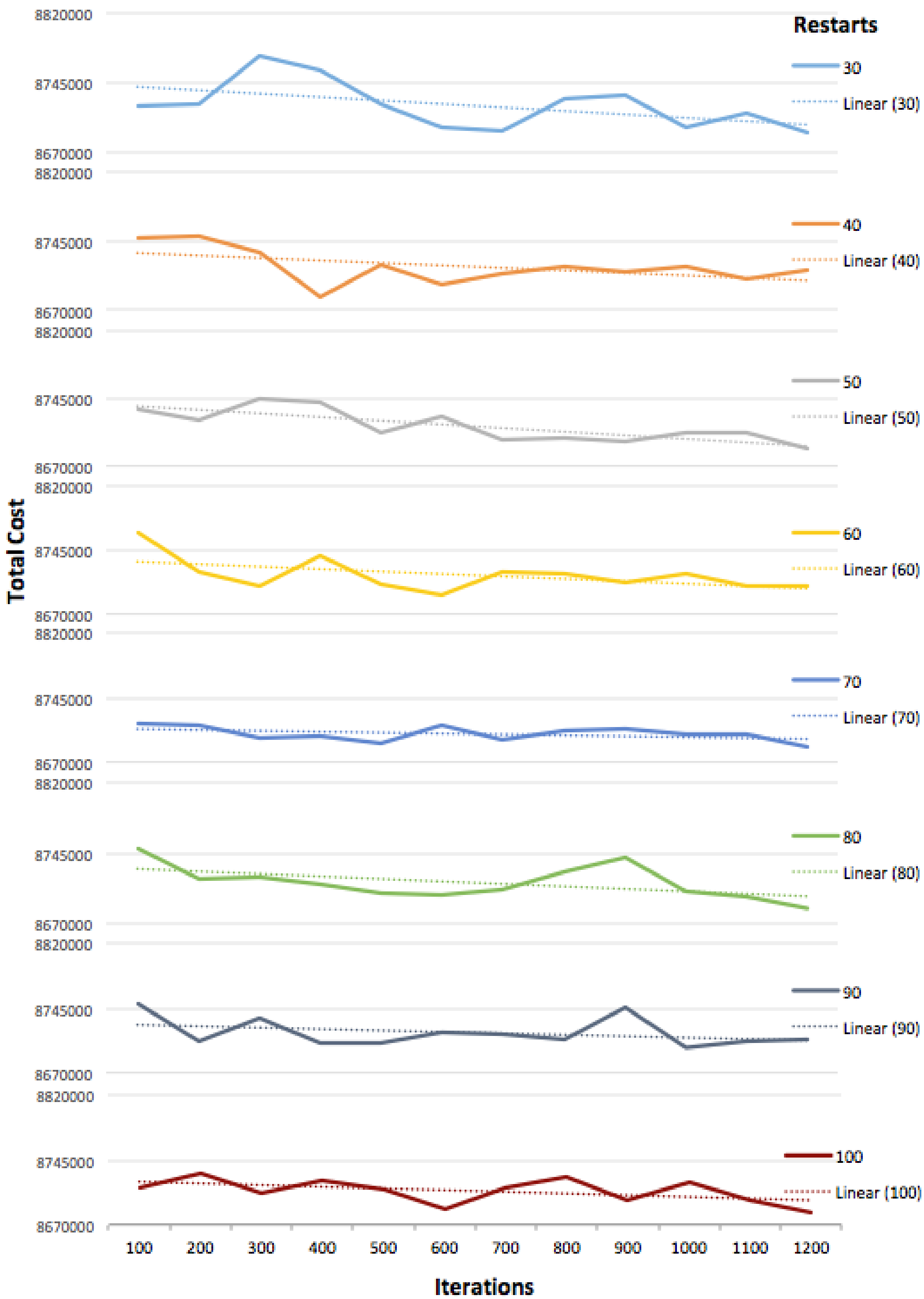


Figure 7.17: Evolution of the total cost with respect to the number of iterations for the LNS with flexible departures

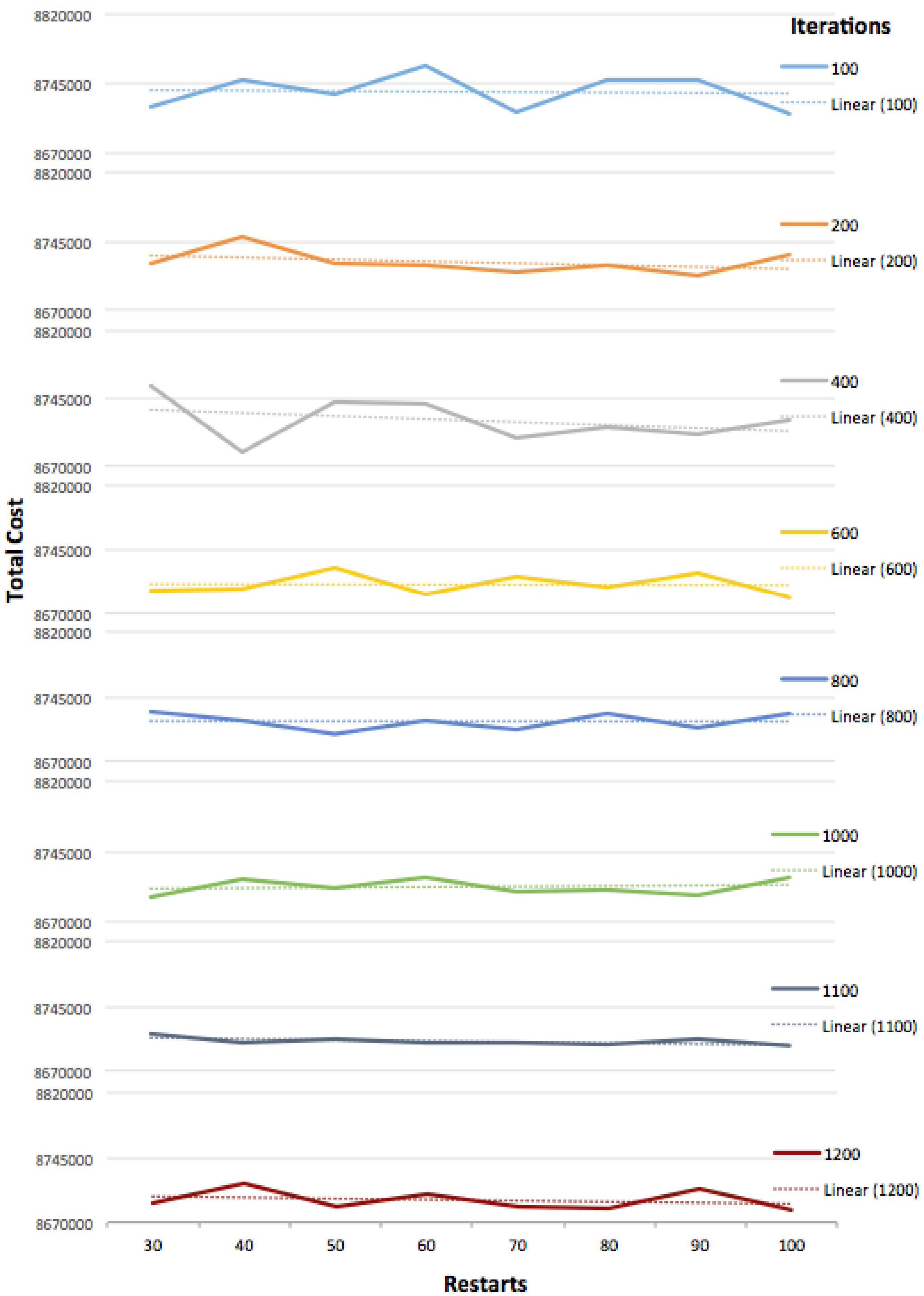


Figure 7.18: Evolution of the total cost with respect to the number of restarts for the LNS with flexible departures

dependence of the costs on the number of restarts compared to LNS with flexible departures (see Figure 7.20).

The lowest solution cost for LNS with flexible departures (NOK 8683993.67) is obtained for the scenario with maximum number of restarts and iterations (100 restarts and 1200 iterations) and computational time 2:26:56 or 8816 sec. The highest cost is for the scenario with 20 restarts and 100 iterations (NOK 8813507.93) and computational time 0:04:51 or 291 sec. The difference between the lowest and highest cost is NOK 129514 or in other words the decrease from maximum to minimum costs accounted for 1.47% while the time decreased by 8525 sec. or 96.7%. Similarly for the LNS with flexible departures and coupled vessels the minimal cost (8736779.16) is obtained for the scenario with 20 restarts and 100 iterations and computational time 00:10:17 or 617 sec. The lowest cost (8648589.53) is obtained for the scenario with 90 restarts and 800 iterations with computational time 05:27:23 or 19643 sec. We note that minimal cost is not achieved for the scenario with maximum number of restarts and iterations since there is no dependency of costs on the restarts and iterations is (due to randomnesses in the logic of the algorithm). The difference between the highest and lowest costs is NOK 88189.63 or 1.01%. The difference in computational time is 05:17:07 (19026 sec.) or 96.86%.

There is no concrete combination for the number of restarts and iterations. The optimal combination is user-defined based on the conducted analysis. Those who are interested in best results may of course select larger numbers of restarts and iterations and those who are concerned about less computational time may select the minimal defined combinations for both algorithms to guarantee the minimal number of vessels.

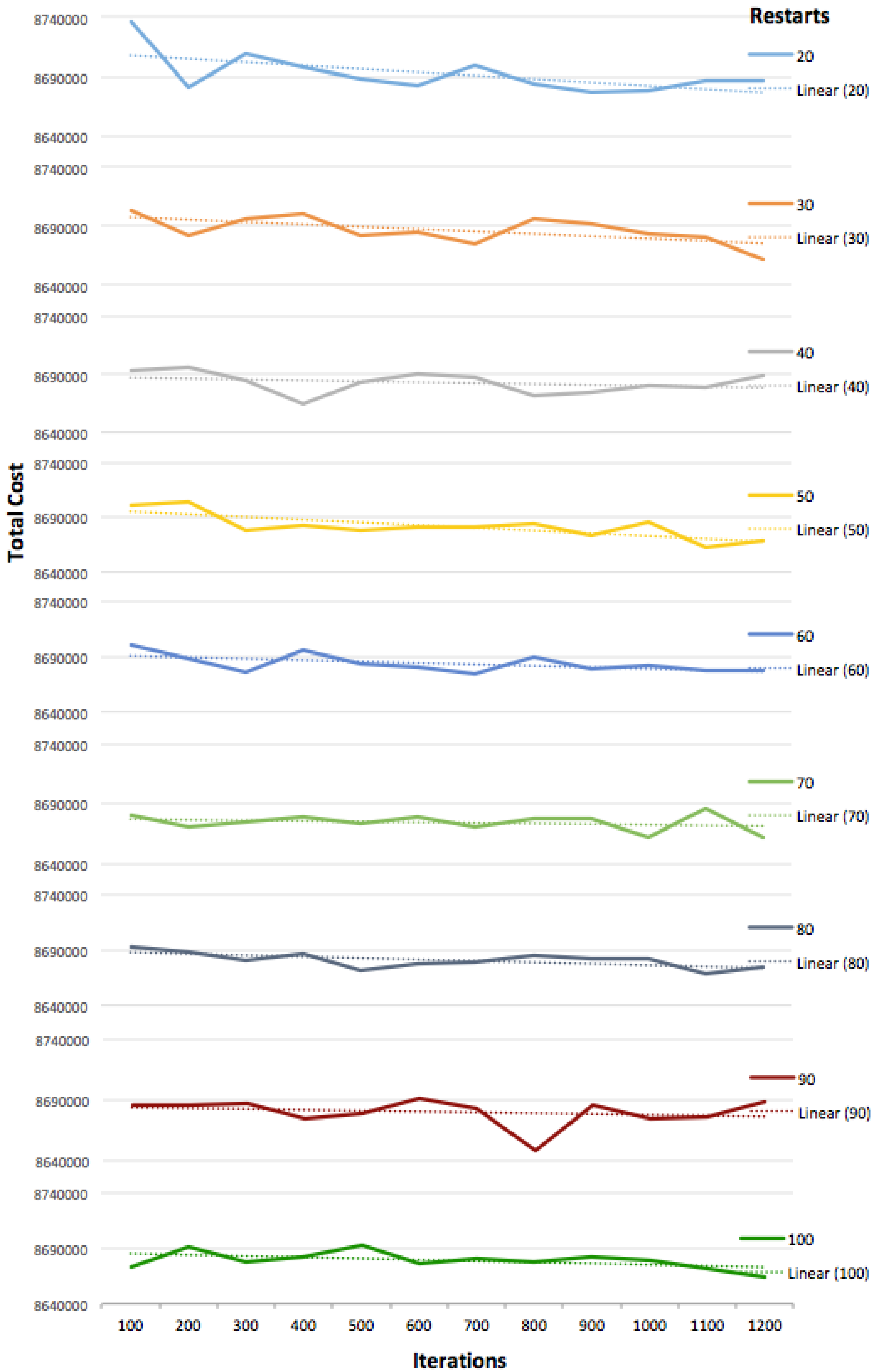


Figure 7.19: Evolution of the total cost with respect to the number of iterations for the LNS with flexible departures and coupled vessels

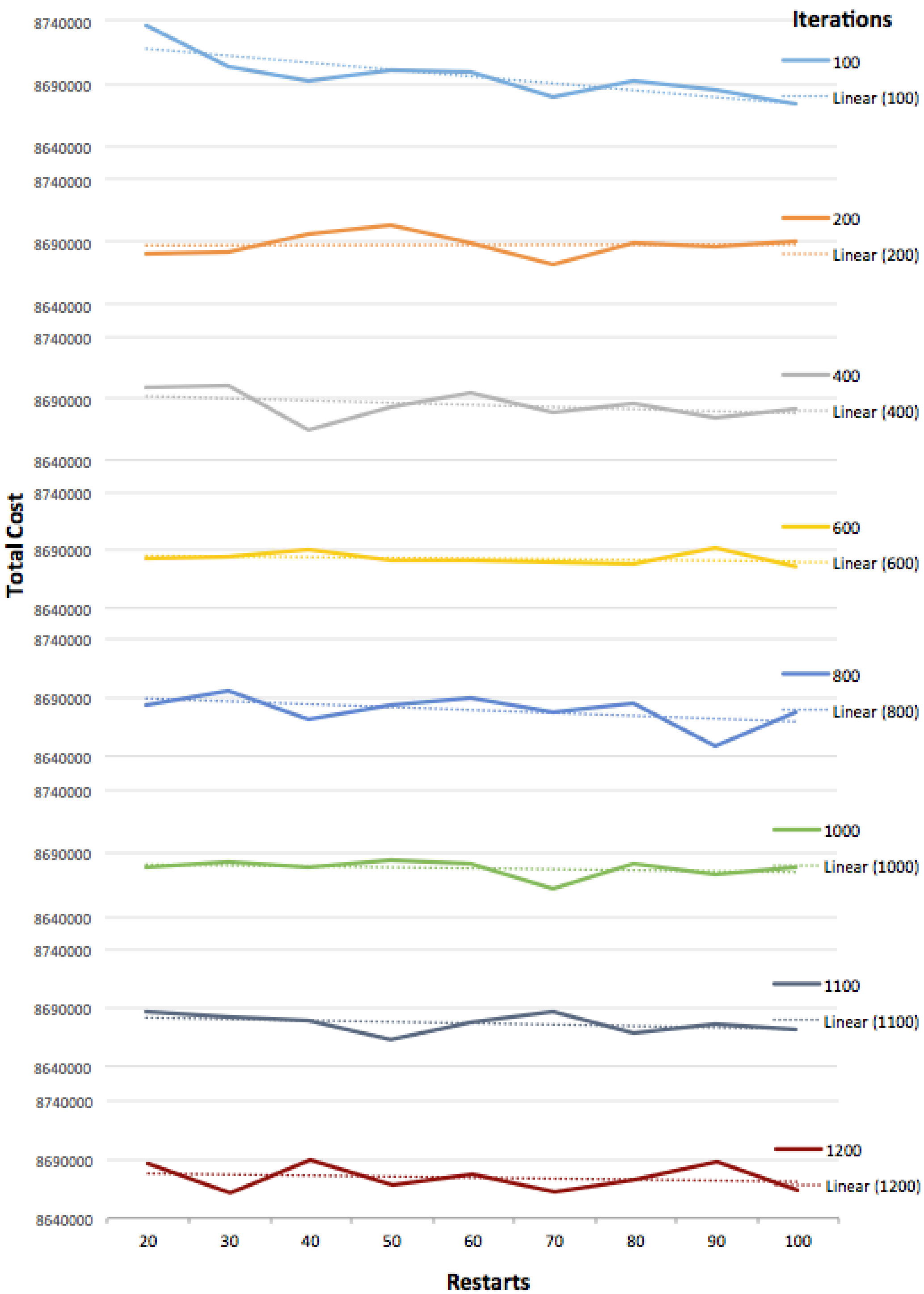


Figure 7.20: Evolution of the total cost with respect to the number of restarts for the LNS with flexible departures and coupled vessels

8 Conclusions and Directions for Further Research

In the upstream offshore petroleum logistics one of the main cost contributors are supply vessels. They are used to deliver materials and equipment from an onshore supply base to drilling rigs and oil platforms. Supply planning of offshore installations is quite crucial process for petroleum business. The down time of an installation, in case of disrupt or delay of supply, is extremely costly and therefore sufficient number of supply vessels is required to serve all installations in a due time. On the other hand, vessel charter and fuel costs are quite high as well. For this reason, efficient planning of supply vessels is required to cut supply costs while maintaining high service level.

In this thesis, we deal with a challenging real-life problem of supply vessel planning in the upstream offshore petroleum logistics. The problem represents an extension of the well-known periodic supply vessel planning problem (PSVPP). The problem is tactical with one week planning horizon. Extensions are implemented with the aim of the fleet size and fuel costs reduction. The classical PSVPP implies departure of a vessel for a voyage at a single and fixed point of time during the day. We extended the problem to "flexible departures" when there are several possible departure times during the day and actual departure time is a decision variable. Another extension involves the concept of coupled vessels when any two vessels can sail each other voyages on the subsequent weeks. Coupled vessels are used to eliminate possible schedule infeasibility in case of "end-of-week" effects when a vessel has a voyage starting at the end of one week and ending at the beginning of the next (therefore first and last voyages may overlap in time).

The objective of the research is to create a decision support tool for periodic supply vessel planning problem with flexible departures and coupled vessels that is able to solve problems of large size within reasonable time.

Within this thesis, we analyzed existing methods of routes planning regarding the studied problem. As a solution method, we selected the large neighbourhood search (LNS) heuristic for constructing weekly periodic vessels schedules and modified it according to the specified extensions, which were implemented in two steps. First, we implemented flexible departure times and after, the concept of coupled vessels.

For validation and performance assessment and of the heuristic, we generated a set of instances using real data for up to 26 installations and conducted a set of experiments. For heuristic validation, we developed two-phase method with flexible departures and compared the results on small and medium instance sizes, since the method is able to solve instances of limited size. For small and medium size instances heuristic shows optimal or near optimal results while computational time compared to the two-phase approach is extremely shorter. For large size instances we first compared results of the heuristics with fixed and flexible departures to assess efficiency of implementation of flexible departures. The average cost reduction accounted for 4,12% and for several instances the fleet size was reduced, while computation time increased for several minutes. Further we compared results of the heuristic with flexible departures to the heuristic with flexible departures and coupled vessels where the average cost reduction accounted for 0,45% at the expense of doubling the computational time.

Since, heuristic is applied for a number of restarts and iterations within each restart, which are defined by a user. We also conducted a set of experiments to assess how changes the heuristic's performance with the increase of the number of restarts and iterations. Experiments show that to get relatively good solution it is enough to run the algorithm at least for 20 restarts and 200 iterations.

The developed tool could be a good framework for practitioners, while dealing with tactical supply vessels planning since experiments showed its efficiency and high speed. Furthermore, the output of the algorithm is provided in a user-friendly form so that any additional analysis or post-processing are not required. And of course the tool can be used by researches for the analysis of the heuristics under different conditions and for its further extension and performance improvement.

Given the conclusion we draw from our computational study we can suggest the following directions for future research:

- In this thesis we introduce the concept of coupled vessels together with its benefits. As one can see it is pretty straight forward, it relaxes the constraints and than tries to find a better solution with a use of coupled vessels, but it doesn't know whether coupled vessels exist or not. Hence, to assess the effectiveness of the chosen approach, future research should consider some smart techniques to find a pair to the vessel, which voyages overlaps. While searching for a possible pair, additional procedures, such as movements of voyages and visits may be considered in order to free up some space at the beginning of the week;

- A critical issue in design of the neighbourhood search heuristic is the choice of the neighbourhood structure or in other words, how the neighbourhood is defined. Than larger the neighbourhood, than longer it takes to search new neighbourhood at each restart. Generally it might take many runs of the algorithm at different starting points to find a new neighbourhood. We have introduced a remove-insert visits procedure as one of them to search neighbourhoods, it involves quite a lot operations which are based on a random choice. So the future research might be based on the effectiveness of chosen approach and probably more effective heuristic might be used in order to find new neighbourhoods (see Ahujaa et al. 2000; Mairy et al. 2000; Pisinger et al. 2006, 2007);

- We have considered only single base PSVPP in this thesis, all presented solution algorithms could be further easily extended into a multi-base PSVPP, where vessels fleet is shared between all supply bases, thus vessels are better utilized;

- To better evaluate the significance of flexible departures and coupled vessels, conduct additional computations on the problem with higher demands and heterogeneous vessel fleet (Such conditions were tested, during the development of the algorithms, but experiments conducted in this thesis ignore demands for installations and vessels fleet is assumed to be homogeneous, since the data provided by Statoil didn't contain that information).

References

Aas, Bjø rnar, Irina Gribkovskaia, Halskau Ø yvind, and Alexander Shlopak. 2007. "Routing of supply vessels to petroleum installations". *Int. J. Phys. Distrib. Logist. Manag.* 37 (2): 164–179.

Ahujaa, Ravindra K, James B Orlinb, Dushyant Sharmac, R.K. Ahuja, J.B. Orlin, and D. Sharma. 2000. "Very large-scale neighborhood search". *Int. Trans. Oper. Res.* 7 (4-5): 301–317. `http://www.blackwell-synergy.com/doi/abs/10.1111/j.1475-3995.2000.tb00201.x`.

Brownlee, J. 2011. *Clever Algorithms: Nature-inspired Programming Recipes.* LULU Press. `https://books.google.no/books?id=QOhlAgAAQBAJ`.

Christiansen, Marielle, Kjetil Fagerholt, Bjø rn Nygreen, and David Ronen. 2007. "Maritime Transportation". *Handbooks Oper. Res. Manag. Sci.* 14 (C): 189–284.

E.G. Coffman, Jr., M R Garey, and D S Johnson. 1984. "Approximation Algorithms for Bin-Packing – An Updated Survey". In *Algorithm Des. Comput. Syst. Des.* Ed. by G Ausiello, M Lucertini, and P Serafini, 284:49–106. International Centre for Mechanical Sciences. Springer Vienna. `http://dx.doi.org/10.1007/978-3-7091-4338-4\_3`.

Fagerholt, Kjetil, and Hakon Lindstad. 2000. "Optimal policies for maintaining a supply service in the Norwegian Sea". *Omega* 28 (3): 269–275.

Fourer, R., D. M. Gay, and B. W. Kernighan. 1990. "A Modeling Language for Mathematical Programming". *Manage. Sci.* 36 (5): 519–554.

Gribkovskaia, I, G Laporte, and a Shlopak. 2008. "A tabu search heuristic for a routing problem arising in servicing of offshore oil and gas platforms". *J. Oper. Res. Soc.* 59 (11): 1449–1459.

Halvorsen-Weare, Elin E., Kjetil Fagerholt, Lars Magne Nonå s, and Bjorn Egil Asbjornslett. 2012. "Optimal fleet composition and periodic routing of offshore supply vessels". *Eur. J. Oper. Res.* 223 (2): 508–517.

Halvorsen-Weare, Elin E., and Kjetil Fagerholt. 2011. "Robust Supply Vessel Planning". In *Netw. Optim. SE - 62*, ed. by Julia Pahl, Torsten Reiners, and Stefan Voß, 6701:559–573. Lecture Notes in Computer Science. Springer Berlin Heidelberg. http://dx.doi.org/10.1007/978-3-642-21527-8_62.

Hart, WilliamE. 2009. "Python Optimization Modeling Objects (Pyomo)". In *Oper. Res. Cyber-Infrastructure SE - 1*, 47:3–19. Operations Research/Computer Science Interfaces. Springer US. http://dx.doi.org/10.1007/978-0-387-88843-9_1.

Mairy, Jean-baptiste, Pierre Schaus, and Yves Deville. "Generic Adaptive Heuristics for Large Neighborhood Search": 1–15.

Norlund, Ellen Karoline, and Irina Gribkovskaia. 2013. "Reducing emissions through speed optimization in supply vessel operations". *Transp. Res. Part D Transp. Environ.* 23 (x): 105–113. http://dx.doi.org/10.1016/j.trd.2013.04.007.

Pisinger, David, and Stefan Ropke. 2007. "A general heuristic for vehicle routing problems". *Comput. Oper. Res.* 34 (8): 2403–2435.

— . 2006. "An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows". *Transp. Sci.* 40 (4): 455–472.

— . 2010. "Large Neighborhood Search". In *Handb. Metaheuristics SE - 13*, 146:399–419. International Series in Operations Research & Management Science. Springer US. http://dx.doi.org/10.1007/978-1-4419-1665-5_13.

Shaw, Paul. 1998. "Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems". *Princ. Pract. Constraint Program. 1998*: 417–431.

Shyshou, Aliaksandr, Irina Gribkovskaia, Gilbert Laporte, and Kjetil Fagerholt. 2012. "A Large Neighbourhood Search Heuristic for a Periodic Supply Vessel Planning Problem Arising in Offshore Oil and Gas Operations". *INFOR* 50 (4): 195–204. http://utpjournals.press/doi/10.3138/infor.50.4.195.

Statoil ASA. 2015. *Statoil Annual Report on Form 20-F*. Vol. 15. d. http://www.statoil.com/no/InvestorCentre/AnnualReport/AnnualReport2014/Documents/DownloadCentreFiles/01_KeyDownloads/20-F_2014.pdf.

The Ministry of Petroleum and Energy and Norwegian Petroleum Directorate. 2014. "Facts 2014 - The Norwegian Petroleum Sector". http://www.regjeringen.no/upload/OED/pdffiler/Faktaheftet/Fakta2014OG/Facts_2014_nett.pdf.

Appendices

A. Evaluation of the cost and computation time for the LNS with flexible departures (instance with 26 installations)

Table A.1: A heat map of the number of vessels used for the LNS with flexible departures with respect to the number of restarts and iterations

		Restarts													
		1	3	5	10	15	20	30	40	50	60	70	80	90	100
Iterations	10	7	6	6	6	6	6	6	6	6	6	5	6	5	6
	20	7	6	6	6	6	6	6	5	5	6	6	5	5	5
	30	6	6	6	6	5	5	5	5	5	6	5	6	5	6
	40	7	6	6	6	6	6	5	6	5	5	6	5	6	5
	50	7	6	6	6	6	6	5	5	5	6	5	5	6	5
	60	6	6	6	6	6	6	6	5	5	5	6	5	5	5
	70	6	6	6	6	6	6	5	5	5	5	6	6	6	5
	80	6	6	6	6	6	5	5	5	5	6	5	5	6	5
	90	6	6	6	6	5	5	5	5	5	5	5	5	5	5
	100	6	5	6	6	6	5	5	5	5	5	5	5	5	5
	200	6	6	6	5	5	5	5	5	5	5	5	5	5	5
	300	6	6	5	5	5	5	5	5	5	5	5	5	5	5
	400	6	6	5	5	5	5	5	5	5	5	5	5	5	5
	500	6	6	5	6	5	5	5	5	5	5	5	5	5	5
	600	6	6	5	5	5	5	5	5	5	5	5	5	5	5
	700	6	6	5	5	5	5	5	5	5	5	5	5	5	5
	800	7	5	5	6	5	5	5	5	5	5	5	5	5	5
	900	6	6	5	6	5	5	5	5	5	5	5	5	5	5
	1000	5	6	6	5	5	5	5	5	5	5	5	5	5	5
	1100	6	5	5	5	5	5	5	5	5	5	5	5	5	5
	1200	6	5	5	6	6	5	5	5	5	5	5	5	5	5

Table A.2: A heat map of the cost and computation time for the LNS with flexible departures with respect to the number of restarts and iterations

Number of LNS Restarts → **and iterations ↓**		**1**	**3**	**5**	**10**	**15**	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**
Obj. value	10	11575783	10118727	10131102	10163470	10145201	10127986	10114062	10108399	10114563	10118200	8718329	10095125	8772529	10125793
	20	11616865	10183249	10204427	10138564	10142061	10121890	10095750	8735806	8772408	10118261	10112436	8747355	8735804	8752968
	30	10201359	10147901	10154910	10102314	8796820	8717729	8750481	8749063	8716700	10107643	8753714	10111085	8821274	10106988
	40	11598462	10176028	10158735	10136469	10118780	10124456	8802012	10121760	8699417	8787169	10111906	8762918	10098922	8779896
	50	11587007	10154496	10115516	10121318	10121658	10097066	8751844	8766419	8727783	10107164	8754033	8733035	10086004	8758382
	60	10168336	10148517	10140232	10091604	10106677	10110564	10101439	8797238	8700826	8739578	10106830	8755471	8726676	8735346
	70	10191448	10139248	10151363	10105419	10135557	10107842	8740982	8708034	8732523	8726184	10103268	10101065	10094596	8722014
	80	10161357	10131217	10140732	10118068	10128616	8743871	8727960	8785530	8746617	10093208	8743590	8739859	10095140	8763336
	90	10196598	10137451	10128397	10133966	8728869	8719710	8776688	8799709	8724426	8739799	8723666	8710844	8759781	8758280
	100	10200519	8774155	10169725	10133905	10116860	8813508	8719752	8748907	8733782	8765050	8714304	8750333	8749650	8713103
	200	10160417	10115461	10149498	8777227	8753723	8711360	8722458	8751575	8720624	8719051	8712656	8718764	8707270	8730307
	300	10196853	10155474	8735636	8707206	8768536	8759206	8775328	8733343	8744151	8703233	8697968	8719366	8733115	8705779
	400	10245789	10177113	8715856	8728501	8711903	8720612	8758581	8684381	8741824	8738373	8700790	8712937	8705358	8720843
	500	10108108	10116127	8754455	10084067	8746135	8696430	8722030	8718850	8707911	8705260	8691027	8703429	8704093	8711637
	600	10167547	10123959	8763545	8726500	8720027	8714080	8696872	8699096	8724426	8691596	8713300	8700417	8717136	8688739
	700	10134697	10144958	8717931	8771693	8753963	8735475	8694627	8710182	8698698	8720719	8695516	8705814	8714640	8712835
	800	11516492	8752841	8761104	10100530	8715104	8724552	8728039	8717831	8701346	8716752	8706997	8725122	8708276	8725380
	900	10156471	10112631	8735997	10102441	8695804	8723090	8732746	8711708	8698068	8707473	8708112	8740698	8745980	8699330
	1000	8711712	10097257	10103026	8700321	8726537	8724652	8696660	8716769	8706173	8717845	8703368	8705553	8699238	8718335
	1100	10127659	8703744	8701139	8773192	8751006	8727404	8712365	8703137	8706118	8702600	8701505	8699651	8705920	8698901
	1200	10099167	8726371	8741646	10093748	10086998	8725637	8691989	8714641	8688344	8702168	8687236	8686527	8709301	8683994
Time	10	00:00:05	00:00:13	00:00:21	00:00:33	00:00:50	00:01:08	00:01:28	00:02:24	00:02:49	00:02:42	00:04:23	00:04:15	00:05:14	00:05:16
	20	00:00:02	00:00:09	00:00:32	00:00:46	00:01:07	00:01:26	00:02:07	00:03:13	00:02:42	00:03:14	00:05:40	00:06:12	00:06:51	00:08:05
	30	00:00:02	00:00:19	00:00:27	00:00:59	00:01:27	00:01:49	00:03:43	00:03:35	00:03:59	00:04:21	00:07:11	00:07:01	00:09:02	00:10:08
	40	00:00:01	00:00:18	00:00:22	00:01:13	00:01:37	00:02:24	00:04:27	00:04:07	00:04:26	00:05:14	00:07:12	00:08:58	00:09:16	00:09:47
	50	00:00:10	00:00:36	00:00:49	00:01:31	00:02:10	00:01:59	00:03:41	00:05:23	00:05:11	00:05:23	00:09:01	00:10:13	00:11:37	00:12:30
	60	*00:00:30*	*00:00:23*	*00:00:34*	*00:01:08*	00:02:25	00:02:41	00:04:25	00:04:46	00:05:26	00:06:30	00:11:39	00:11:50	00:12:09	00:13:29
	70	00:00:03	00:00:31	00:00:55	00:01:51	00:02:06	00:03:14	00:03:25	00:06:10	00:07:02	00:06:46	00:09:59	00:12:52	00:13:24	00:15:24
	80	00:00:13	00:00:21	00:00:58	00:01:43	00:02:33	00:03:06	00:04:46	00:07:02	00:08:21	00:06:54	00:12:45	00:13:40	00:15:55	00:16:52
	90	00:00:15	00:00:47	00:00:46	00:02:23	00:02:54	00:04:00	00:05:58	00:06:57	00:09:19	00:08:09	00:12:29	00:14:31	00:16:27	00:19:06
	100	00:00:03	00:00:28	00:00:31	00:01:51	00:04:03	00:04:51	00:06:19	00:08:34	00:07:49	00:09:31	00:10:12	00:15:31	00:12:29	00:20:29
	200	00:00:21	00:01:41	00:01:02	00:02:53	00:04:04	00:05:56	00:10:12	00:12:00	00:15:03	00:15:30	00:16:37	00:26:53	00:23:45	00:33:44
	300	00:00:36	00:01:30	00:01:44	00:04:53	00:05:38	00:08:08	00:12:43	00:18:03	00:18:16	00:19:46	00:24:56	00:36:25	00:30:55	00:48:03
	400	00:00:12	00:00:47	00:02:02	00:05:37	00:07:28	00:11:52	00:12:45	00:18:06	00:26:47	00:27:32	00:29:51	00:47:53	00:38:37	00:55:47
	500	00:01:16	00:02:02	00:03:23	00:05:54	00:11:06	00:12:28	00:21:56	00:29:51	00:25:42	00:27:32	00:42:55	00:54:03	00:44:10	01:14:04
	600	00:00:31	00:02:20	00:04:08	00:06:14	00:11:42	00:12:43	00:26:52	00:23:52	00:32:12	00:31:35	00:41:42	01:07:43	00:55:54	01:20:56
	700	00:01:11	00:02:44	00:06:03	00:10:26	00:11:36	00:17:11	00:23:48	00:33:40	00:34:17	00:39:50	00:48:27	01:14:44	01:08:59	01:39:18
	800	00:00:17	00:02:59	00:06:49	00:09:29	00:15:11	00:25:18	00:26:23	00:39:09	00:35:54	01:02:34	00:56:59	01:18:09	01:07:15	01:45:02
	900	00:01:06	00:04:13	00:04:36	00:08:18	00:18:44	00:22:17	00:24:13	00:34:38	01:20:27	00:50:00	00:59:18	01:25:21	01:15:05	01:45:11
	1000	00:00:55	00:04:05	00:06:42	00:06:34	00:18:22	00:23:19	00:31:47	00:47:09	00:46:41	00:57:12	01:04:03	02:02:23	01:26:56	02:05:18
	1100	00:02:07	00:04:04	00:08:27	00:12:19	00:19:59	00:22:07	00:46:16	00:56:38	00:50:44	01:00:22	01:22:05	01:53:48	01:29:03	02:00:00
	1200	00:02:51	00:04:00	00:03:32	00:10:56	00:21:17	00:31:26	00:37:48	01:01:06	01:06:42	01:21:24	01:07:25	01:56:21	01:34:19	02:26:56

B. Evaluation of the cost and computation time for the LNS with flexible departures and coupled vessels (instance with 26 installations)

Table B.1: A heat map of the number of vessels used for the LNS with flexible departures and coupled vessels with respect to the number of restarts and iterations

		Restarts													
		1	3	5	10	15	20	30	40	50	60	70	80	90	100
Iterations	10	7	6	6	6	6	6	6	5	5	6	5	6	5	6
	20	7	6	6	6	5	5	6	5	5	6	6	5	5	5
	30	6	6	5	6	5	5	5	5	5	6	5	6	5	6
	40	7	6	6	6	5	6	5	6	5	5	6	5	6	5
	50	6	6	6	5	6	5	5	5	5	5	5	5	6	5
	60	6	6	5	6	5	6	6	5	5	5	5	5	5	5
	70	6	6	5	5	5	6	5	5	5	5	5	5	6	5
	80	6	5	6	6	6	5	5	5	5	6	5	5	5	5
	90	6	6	6	6	5	5	5	5	5	5	5	5	5	5
	100	6	5	6	5	6	5	5	5	5	5	5	5	5	5
	200	6	6	5	5	5	5	5	5	5	5	5	5	5	5
	300	6	6	5	5	5	5	5	5	5	5	5	5	5	5
	400	6	6	5	5	5	5	5	5	5	5	5	5	5	5
	500	5	6	5	6	5	5	5	5	5	5	5	5	5	5
	600	6	6	5	5	5	5	5	5	5	5	5	5	5	5
	700	6	6	5	5	5	5	5	5	5	5	5	5	5	5
	800	7	5	5	6	5	5	5	5	5	5	5	5	5	5
	900	6	6	5	6	5	5	5	5	5	5	5	5	5	5
	1000	5	6	6	5	5	5	5	5	5	5	5	5	5	5
	1100	6	5	5	5	5	5	5	5	5	5	5	5	5	5
	1200	6	5	5	6	5	5	5	5	5	5	5	5	5	5

Table B.2: A heat map of the cost and computation time for the LNS with flexible departures and coupled vessels with respect to the number of restarts and iterations

Number of LNS Restarts → and iterations ↓		1	3	5	10	15	20	30	40	50	60	70	80	90	100
Obj. value	10	11571654	10101885	10119748	10129200	10100121	10089571	10083374	8695992	8716421	10091441	8685120	10071362	8726511	10091657
	20	11574163	10155452	10126191	10112865	8726925	8701626	10075261	8699541	8695751	10083927	10085239	8722568	8704338	8689815
	30	10179624	10141036	8800670	10091647	8723018	8684122	8701018	8713785	8690490	10073981	8715384	10074767	8709914	10078685
	40	11596440	10139434	10136970	10126270	8702647	10103385	8716595	10078172	8690934	8712054	10081472	8712910	10084133	8708107
	50	10130294	10133573	10098286	8719679	10079436	8702917	8710584	8732607	8673546	8712456	8690061	8705476	10077865	8687702
	60	10126657	10125180	8768821	10084104	8754669	10088361	10072019	8732822	8684671	8691175	8748785	8714722	8700196	8686184
	70	10187416	10106530	8716884	8747272	8724827	10101616	8707109	8668183	8696988	8686224	8672993	8714115	10068386	8678154
	80	10139088	8729870	10104392	10087420	10103060	8698709	8692886	8699125	8688875	10072390	8685118	8687594	8713606	8700627
	90	10123837	10108584	10107078	10119745	8707977	8682999	8730411	8703202	8679982	8696790	8699115	8688969	8707243	8688265
	100	10198652	8740877	10111357	8800439	10096823	8736779	8703328	8693283	8700765	8700149	8680207	8692237	8685749	8673722
	200	10143347	10084994	8685381	8725364	8701587	8680880	8682659	8695995	8703925	8688540	8671524	8688898	8685878	8691319
	300	10122778	10103876	8728566	8692930	8720311	8709646	8696338	8685083	8678466	8675873	8674622	8681257	8687167	8678001
	400	10209312	10099219	8711066	8711546	8686849	8698820	8701266	8664503	8683193	8695120	8678960	8686903	8675148	8682006
	500	8731871	10091530	8710659	10075954	8697221	8688407	8682093	8683667	8678560	8682930	8673425	8672114	8679426	8692361
	600	10159500	10112384	8690978	8701460	8689249	8682251	8684494	8691071	8681758	8681077	8679073	8677531	8691144	8675676
	700	10119609	10107707	8689141	8704161	8713825	8700316	8674990	8688221	8680394	8673728	8671111	8679632	8682994	8680438
	800	11515337	8720296	8702985	10076190	8700134	8683921	8695991	8672052	8683993	8689840	8678130	8684656	8648590	8677453
	900	10113410	10103468	8703294	10086063	8689218	8677349	8691545	8673860	8682557	8679349	8677742	8681479	8686006	8681960
	1000	8707339	10093599	10075186	8689668	8688365	8678961	8683430	8679744	8685495	8681702	8662471	8682565	8674008	8679078
	1100	10126592	8686565	8690599	8707432	8696339	8686906	8681304	8678934	8662693	8677611	8686352	8667876	8676536	8672108
	1200	10098939	8705158	8691413	10088868	8754173	8687193	8661457	8689200	8668386	8677969	8662471	8679248	8688287	8663453
Time	10	00:00:08	00:00:23	00:00:46	00:01:10	00:01:35	00:03:21	00:06:16	00:06:16	00:04:19	00:08:31	00:08:35	00:15:02	00:09:34	00:15:31
	20	00:00:06	00:00:21	00:00:49	00:02:11	00:02:34	00:03:01	00:14:33	00:07:13	00:17:01	00:12:00	00:21:37	00:13:02	00:18:18	00:15:46
	30	00:00:08	00:00:38	00:01:17	00:02:29	00:02:34	00:05:34	00:07:55	00:09:45	00:13:01	00:17:30	00:15:00	00:18:46	00:19:25	00:41:32
	40	00:00:09	00:00:43	00:01:21	00:03:03	00:04:20	00:07:08	00:07:51	00:10:36	00:16:06	00:11:46	00:35:57	00:27:01	00:20:56	00:22:18
	50	00:00:28	00:01:32	00:01:56	00:04:42	00:05:29	00:08:28	00:09:52	00:17:48	00:12:36	00:30:42	00:24:00	00:25:31	00:52:12	00:32:01
	60	*00:00:43*	*00:01:03*	*00:01:51*	*00:05:22*	00:06:45	00:15:39	00:11:31	00:24:18	00:28:00	00:15:08	00:44:06	00:30:12	00:49:22	00:52:52
	70	00:00:12	00:01:02	00:01:56	00:04:42	00:06:17	00:10:06	00:09:33	00:22:31	00:17:55	00:21:42	00:31:02	01:03:53	01:10:14	01:04:55
	80	00:00:24	00:01:23	00:02:12	00:05:06	00:07:15	00:10:14	00:12:22	00:13:37	00:31:38	00:24:42	00:41:18	00:55:46	01:04:22	00:42:32
	90	00:00:36	00:01:25	00:02:22	00:08:35	00:06:42	00:11:24	00:12:50	00:14:25	00:34:47	00:19:31	00:18:46	00:44:48	00:42:52	00:37:13
	100	00:00:12	00:01:01	00:03:11	00:06:30	00:08:50	00:10:17	00:17:11	00:23:32	00:19:42	00:19:17	00:36:00	00:46:33	00:35:10	00:57:48
	200	00:01:03	00:04:10	00:03:23	00:06:34	00:11:28	00:23:21	00:30:26	00:25:57	00:24:43	00:56:12	00:46:39	01:21:50	01:23:40	02:06:58
	300	00:01:34	00:03:13	00:11:06	00:21:45	00:12:48	00:15:21	00:29:20	00:47:10	00:42:15	00:50:56	02:17:17	01:26:07	01:39:41	06:17:44
	400	00:00:21	00:02:28	00:12:37	00:20:49	00:33:40	00:28:02	00:29:25	01:04:34	00:52:16	01:00:54	02:07:09	01:25:44	01:59:08	02:31:22
	500	00:02:18	00:11:02	00:08:03	00:49:18	00:24:28	00:53:22	00:51:52	01:10:32	01:04:14	02:25:09	03:14:23	03:06:20	05:50:36	06:38:44
	600	00:01:32	00:04:38	00:09:24	00:15:59	00:43:20	00:33:01	00:58:17	02:06:53	01:14:48	04:49:49	01:54:58	13:51:15	04:45:13	07:15:38
	700	00:02:02	00:05:18	00:11:56	00:21:21	00:27:03	00:48:29	01:21:15	01:53:27	01:46:08	02:32:34	03:23:38	07:48:19	02:28:33	04:46:40
	800	00:00:49	00:06:18	00:11:59	00:32:52	00:49:19	00:54:20	01:18:35	02:21:31	03:56:09	02:14:35	02:43:19	15:09:58	05:27:23	05:56:44
	900	00:02:19	00:07:52	00:13:28	00:45:30	01:45:15	00:51:45	01:03:39	03:10:21	02:18:52	03:25:22	03:28:18	03:17:22	03:01:19	10:22:40
	1000	00:02:20	00:10:44	00:17:26	00:50:17	00:55:27	00:59:31	02:02:19	02:53:04	02:56:09	02:39:15	02:50:00	10:10:08	07:07:13	04:45:12
	1100	00:03:45	00:23:00	00:29:24	00:28:53	00:45:09	01:39:39	01:38:59	01:48:19	02:54:56	05:35:04	04:21:41	12:08:39	08:05:09	06:26:56
	1200	00:10:05	00:08:48	00:12:00	01:17:04	01:24:26	01:07:26	03:34:40	03:35:42	03:23:17	05:32:29	02:50:00	13:22:32	06:40:22	09:39:28