A variable neighborhood search heuristic for disruption management in offshore oil and gas logistics

Magnus Stålhane[a], Nils Albjerk[a], Teodor Danielsen[a], Stian Krey[a], Kjetil Fagerholt[a,b]

[a]*Norwegian University of Science and Technology,*
Department of Industrial Economics and Technology Management
Alfred Getz veg 3, NO-7491,
Trondheim, Norway
[b]*SINTEF Ocean,*
Otto Nielsens veg 10, NO-7052
Trondheim, Norway

Abstract

This paper studies operational planning and disruption management in offshore oil and gas logistics. A significant amount of time is currently spent on operational planning, and major costs are caused by disruptions to the planned routes and schedules for the offshore supply vessels supplying the offshore installations. The disruptions are mainly due to uncertain and harsh weather conditions. To be able to solve real size instances of the planning problem, a variable neighborhood search heuristic is proposed, and tested on instances based on data provided by the case company. The computational results show that the heuristic finds optimal solutions for all the problem instances where the optimal solution is known, and finds high quality solution for larger instances.

Keywords: Disruption management; Heuristics; Offshore supply; Vehicle routing;

1. Introduction

Oil and gas production on the Norwegian continental shelf began more than four decades ago with the discovery of a series of large reservoirs in the North Sea. Today, the North Sea is still the main production area on the Norwegian continental shelf with a total of 60 active oil and gas fields, making Norway one of the biggest oil and gas producers in the world. The industry has generated more than NOK 11 000 billion (approximately 1.3 billion USDs) in present value, and is responsible for close to one third of all value creation in Norway. Even though the production has been ongoing for more than 40 years, it is estimated that 56 % of the total resources have still not been extracted (Regjeringen [17]).

In 1972 the Norwegian state owned company Statoil was founded, and today they are the leading operator on the Norwegian continental shelf with a production of approximately 1 927 million barrels of oil equivalents per day (Statoil [21]). Statoil produces oil and gas at both fixed and floating offshore installations on oil and gas fields in the North Sea, the Norwegian Sea, and the Barents Sea. To perform the activities required at these offshore installations, such as drilling, well operations, and maintenance, different equipment and supplies are needed, and they must

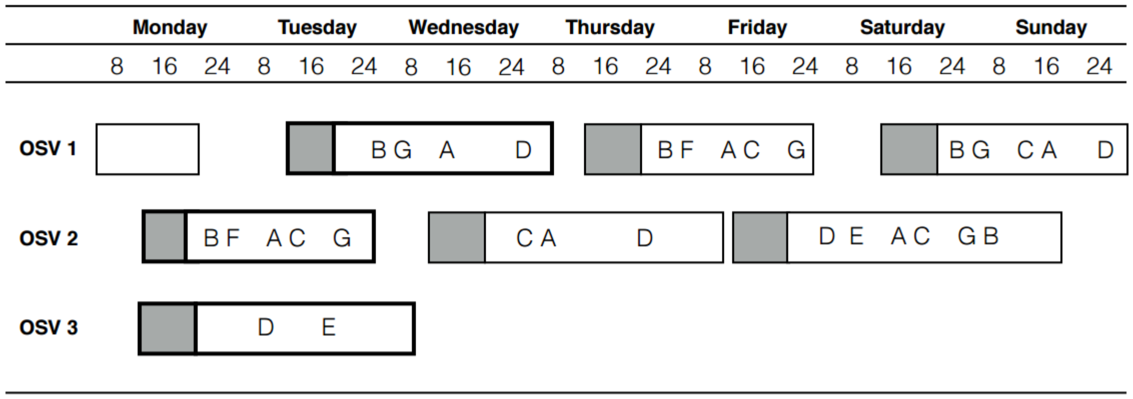

Figure 1: Possible weekly schedule for three OSVs and seven offshore installations (A-G). Each voyage is represented by a rectangle, where the dark square illustrate the time needed for loading at the supply depot. Within each voyage, the order of the letters represent the order in which the offshore installations are visited.

be transported to the installation from one of several onshore supply depots located along the Norwegian coast. Statoil employ what they call the *hold principle*, which entails that all supplies are kept at the onshore depot until a short time before they are needed. This is a measure to keep the amount of stored goods at the installations at a minimal level, since the installations have limited storage capacities.

Supplies are brought to the installations by purpose-built vessels, usually referred to as offshore supply vessels (OSVs). These vessels are very expensive since they are constructed to handle the extreme weather conditions that often occur. The OSVs transport containers and other goods on deck, and bulk cargo in tanks below deck. In addition to supplying the installations, these vessels also carry backload from the installations to the onshore supply depot. This backload consists of waste, used equipment, and mud from the drilling that must be transported back to shore since strict government regulations prohibit any of these materials from being disposed of in the ocean. Costs related to chartering and operating OSVs represent one of the major expenses in the upstream supply chain. In addition, the planning of routes and schedules of the vessels requires significant amounts of time and effort.

Statoil prepare long-term plans for supplying their offshore installations, where a set of voyages are to be sailed on a weekly basis (Statoil [20]). A voyage for an OSV starts at a given onshore supply depot, visits a set of offshore installations in a pre-determined sequence, before returning to the depot. Each voyage normally takes between two and three days and each OSV is scheduled to complete several of these voyages each week. An example of such a plan can be seen in Figure 1, where three OSVs are scheduled to visit seven installations (A-G) during a week. OSVs 1 and 2 each have three voyages each week, while OSV 3 only has one voyage.

However, due to unforeseen events, disruptions of the OSV voyages sometimes occur, which forces the company to re-plan the voyages for some, or all, of the OSVs over the next few days. There are two main sources of disruptions in offshore supply logistics. The first source is the weather conditions, which are notoriously bad in the North Sea, especially during the winter season. Even though the OSVs are built to sustain bad weather, they may have to reduce speed, or in some cases even abort the voyage and return to the supply depot. These delays may cause a knock-on effect on the long-term plan, since the OSV may arrive back at the depot after it should have started its next voyage. In addition, if an OSV has to return to the depot before servicing all the planned installations, the supplies destined for the remaining installations must be brought on subsequent voyages. The second source of disruptions comes from unexpected urgent orders, which is usually a result of equipment failure at an installation that needs spare parts to be fixed. Since a shut-down in production at an installation is extremely costly, these orders always have the highest priority.

To mitigate the effect of these disruptions, the planners at Statoil may deviate from the planned voyages in the wake of a disruption, creating new voyages. However, they also have the possibility to charter an extra OSV from the spot market to handle these disruptions. These types of short term spot charters are very expensive, and a major cost component in the total costs of today's offshore supply chain. Estimates provided by Statoil state that the average short-term charter rate for an OSV is approximately NOK 300,000 per day.

How these disruptions affect the long-term plan and how to mitigate the consequences of these disruptions can be illustrated by an example. Consider the long-term plan shown in Figure 1. For example, suppose that on Monday morning the planners receive information that the weather will be bad for the next few days, and the OSVs must sail at reduced speeds in that period. The consequence of this is that OSV 1 will be delayed arriving back at the depot, since the remainder of its current voyage will be sailed at lower speeds than planned. Aborting voyages is only considered an option when absolutely necessary, since the cargo is already on-board the OSVs and is most likely needed at the platform in the near future (ref. the holding principle). In addition, the next voyages for OSVs 2 and 3, which are to start that Monday afternoon, will also take longer than planned, potentially causing a knock-on effect to the long-term plan. To mitigate the consequences of these disruptions, the voyages marked with a thick border in Figure 1, one for each OSV, may need re-planning. In this re-planning, the goal is to service as many of the planned installations as possible, while ensuring that all OSVs arrive back in time to resume normal operations. For instance, the planners may remove some installations from the next voyage of both OSVs 1 and 2, making them shorter, and thus avoid any knock-on effect to subsequent voyages. OSV 3 has a lot of slack in its schedule, so it is possible to prolong it and add visits to (some of) the installations removed from the voyages of OSVs 1 and 2. If this is not sufficient, a last (and expensive) resort is to charter an extra OSV from the spot market to service some of the installations.

The problem studied in this paper is how to plan new voyages in an operational setting so as to return to the long-term plan as smoothly as possible after a disruption has occurred. The planned operations at the offshore installations may be affected when the service of OSVs deviate from the planned schedule. In addition, significant cost reductions can be made by optimally determining new voyages for the OSVs, given disrupted sailing plans and schedules. Thus, the planning objective is to minimize sailing and chartering costs, while at the same time maintaining a sufficient service level to the offshore installations, and avoiding delays of the OSVs that cause knock-on effects to the long-term plan.

The only previously published paper studying this problem is Albjerk et al. [1], who propose an arc-flow and a path-flow formulation of the problem, together with a method for generating paths. However, these exact solution methods are incapable of solving sufficiently large instances to be of practical use for Statoil. Thus, the main contribution of this paper is to present a variable neighborhood search (VNS) heuristic for this important operational planning problem from offshore oil and gas logistics, where the goal is to minimize the impact of disruptions that often happen to the original plans. The proposed VNS heuristic includes several new problem specific operators, and it is shown that it finds optimal or near-optimal solutions within a reasonable amount of time to a set of real-life instances provided by Statoil. Furthermore, we also discuss the managerial impacts from using a tool based on the proposed heuristic.

The remainder of the paper is structured as follows. Related literature is reviewed in Section 2, while Section 3 provides a formal description of the problem, together with a mathematical formulation. Then, the proposed VNS heuristic is described in Section 4, before the computational experiments are presented in Section 5. Finally, concluding remarks are provided in Section 6.

2. Literature review

In addition to Albjerk et al. [1], who are studying the same problem as in this paper, there are also several other papers in the literature addressing related ship routing and logistics problems from the upstream supply chain for offshore petroleum production. Gribkovskaia et al. [8] study a pickup and delivery problem where offshore oil and gas platforms in the Norwegian Sea are serviced by a single OSV. Unlike the problem studied in this paper, they take into account the available storage space at the offshore installations. To obtain good solutions to the problem with a short computing time, a tabu search heuristic is presented. Sopot and Gribkovskaia [19] extend the problem by taking into account demands for multiple commodities at the offshore installations, and the stowage of these commodities in dedicated compartments on-board the OSVs. They present a mathematical model of the problem and a meta-heuristic to provide high quality solutions in a short amount of time. In contrast to our problem, neither Gribkovskaia et al. [8] nor Sopot and Gribkovskaia [19] consider disruption management.

Halvorsen-Weare et al. [10], Halvorsen-Weare and Fagerholt [9], and Shyshou et al. [18] study the problem of determining the optimal fleet of OSVs to charter in and the weekly routes and schedules for the OSVs (as illustrated in Figure 1). Halvorsen-Weare et al. [10] present an optimization model, currently in use by Statoil, which is based on pre-generation of all possible voyages for all OSVs. Halvorsen-Weare and Fagerholt [9] extend this work by taking into account uncertainties in weather when generating voyages in order to get more robust solutions. Finally, Shyshou et al. [18] present a large neighborhood search heuristic that is capable of finding solutions to larger instances of the same problem. As opposed to the problem discussed in this paper, all these studies consider only outgoing supplies, and not cargo returning to the onshore supply depot, which may become important in a more operational setting.

As shown in Section 3, the problem studied in this paper can be modeled as a pickup and delivery problem (PDP) where all deliveries are transported from a central depot, all pickups are brought back to the central depot, and multiple vehicles are used. Offshore installations might be visited up to two times, either conducting pickup and delivery simultaneously or at different points in time, possibly by different OSVs. We can refer to Berbeglia et al. [3] and Battarra et al. [2] for general surveys on PDPs. Using the classification proposed by Berbeglia et al. [3], the problem can be classified as a 1-M-1|P-D|m, i.e. a one-to-many-to-one pickup and delivery problem with multiple vehicles.

There exists several papers that study PDPs with similarities to our problem, but in different contexts, see for example Dell'Amico et al. [7] for a PDP in reverse logistics, Karlaftis et al. [11] a containership PDP with time deadlines, Polat et al. [15] a PDP in container feeder network design, and Polat et al. [16] a PDP with time limits on the vehicles. All these papers assume that the pickups and deliveries are performed simultaneously and with the same vehicle. Nagy et al. [14] relax this in a similar way as we do in our paper and allow divisible deliveries and pickups. They study the savings that can be achieved by allowing this. However, unlike the above studies, we also treat the time limit of each voyage as a soft, not a hard, constraint, and we have the possibility of chartering an extra vessel.

Relatively few studies regarding disruption management in maritime transportation are encountered, and to the authors' knowledge Albjerk et al. [1] is the only paper addressing disruption management in offshore supply logistics. Brouer et al. [5] consider the vessel schedule recovery problem in container liner shipping. They propose different recovery actions proposed, such as increasing speed, canceling deliveries, and swapping port visits. A model considering costs for sailing, delays, and misplaced cargo is presented, and it is run with data from real life cases. In Kjeldsen [12], a mathematical model for simultaneous rescheduling of ships and cargo in liner shipping is presented. The author mentions poor weather conditions, port congestion, low port productivity, towage, tidal windows, and several other sources of disruptions. Suggested recovery actions are, among others, changing the departure or arrival time at ports, transshipment of cargo between ports, and speed adjustments. The problem is modeled as a ow problem restricted by a time and space network, and is solved by a large neighborhood search heuristic.

The possible measures to handle disruptions in Brouer et al. [5] are mostly similar to the ones available in disruption management in offshore supply. However, both Brouer et al. [5] and Kjeldsen [12] consider container liner shipping, in which there are usually long sailing distances between the ports. The problem of supplying offshore installations from an onshore supply depot involves much shorter distances compared to the container shipping problems discussed above. This implies that a measure such as increasing speed to reduce delays will be less effective than in liner shipping. Since the reason for delays in offshore supply usually also come from poor weather, which restricts the sailing speeds, this also limits the possibilities of increasing speed. The possibility of chartering additional OSVs from the spot market is an additional option available in disruption management in offshore supply. Also, the planning horizons of the container shipping problems are much longer than in the problem discussed in this report.

3. Problem description and mathematical model

The problem studied in this paper is how to minimize the total costs and delays of delivering n cargoes from an onshore depot to a set of offshore installations, while at the same time transporting m cargoes from the installations and back to the onshore depot. The problem can be formulated on a graph $G = (\mathcal{N}, \mathcal{A})$, where the set of nodes $\mathcal{N} = \{0, ..., n+m+1\}$ represent the onshore supply depot (nodes 0 and $n+1$) and the cargoes to be picked up and delivered. The set $\mathcal{N}$ can be divided into the set $\mathcal{N}^P = \{1, ..., n\}$ containing all pickup nodes, and the set $\mathcal{N}^D = \{n+2, ..., n+m+1\}$ containing all delivery nodes. An offshore installation has an associated pickup node if transport of backload cargo to the depot is demanded, and an associated delivery node if delivery of supplies to the installation is demanded. Some installations may have both a pickup and a delivery node associated with it, and we use the term *sibling nodes* to denote two nodes associated with the same installation.

Each node $i \in \mathcal{N}^D$ has a demand D_i for cargo to be delivered, while each node $i \in \mathcal{N}^P$ has a demand P_i for cargo to be picked up. The size of all cargoes are given as the number of square meters it occupies on the deck of an OSV as this is the binding capacity constraint. In addition, each node i has an associated penalty cost C_i^R incurred if the node cannot be serviced on the upcoming set of voyages and must be postponed, depending on the importance and urgency of the associated pickup or delivery cargo.

The set $\mathcal{V} = \{1, ..., k+1\}$ contains all available OSVs, where the OSV $k+1$ represents an OSV chartered from the spot market. Each OSV v has a total deck capacity Q_v measured in square meters, and a cost C_{vij}^S and time T_{vij} (in hours) associated with sailing from node i to node j, and servicing node j. Note that both of these parameters are weather dependent, however, we assume that they are known at the time of planning. Since we are only planning the next voyage for each OSV, i.e. the next couple of days, this is a reasonable assumption. For example, if we know that the weather will be bad in the next couple of days we adjust C^S_{vij} and T_{vij} accordingly.

We assume that we know the time each OSV is available to begin the next voyage, denoted T^{MIN}_v, and the planned departure time of the subsequent voyage for each OSV, denoted T^{MAX}_v. For example, in Figure 1, we may need to re-plan the first voyage of the week for OSV 2, and T^{MAX}_v then corresponds to Wednesday at 8 am, which is when the second voyage is planned to start. However, we do allow the OSV to return back to the depot up to Γ hours after T^{MAX}_v in order for the OSV to serve additional installations, but this incurs a penalty of C^D_v per hour. The spot OSV $k+1$ must be chartered for a whole number of time periods (days), where the number of hours in a time period is denoted by H. The daily time charter rate for the spot OSV is given by C^{TC}.

3.1. Mathematical model

The variable x_{vij} equals 1 if OSV v sails from node i to node j, and 0 otherwise. The auxiliary variable y_{vi} equals 1 if OSV v visits node i, and 0 otherwise. If the visit to node i is postponed (i.e. not serviced on the voyage of any of the OSVs), the variable u_i equals 1, and 0 otherwise. The cargo load variables l_{vij} equals the load measured in square meters on OSV v when sailing from node i to node j. If the arc is not traversed, the corresponding load variable is equal to 0. The number of hours OSV v arrives at the depot after T^{MAX}_v is represented by the variable t^D_v. The number of whole days that the OSV $k+1$ from the spot market needs to be chartered is denoted by t^{TC}. To simplify the notation, the constraints are defined using sets of nodes, even though some constraints may contain combinations of the indicies v, i, and j for which the corresponding variable x_{vij} does not exist (e.g. when $i = j$). These variables can be assumed to take the value 0. Using this notation, the operational planning and disruption management problem can be formulated as follows:

Objective

$$\min \sum_{v\in\mathcal{V}}\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} C^S_{vij}x_{vij} + C^{TC}t^{TC} + \sum_{i\in\mathcal{N}} C^R_i u_i + \sum_{v\in\mathcal{V}\setminus\{k+1\}} C^D_v t^D_v \tag{1}$$

subject to:

$$\sum_{i\in\mathcal{N}\setminus\{0\}} x_{v0i} = 1 \qquad v\in\mathcal{V} \tag{2}$$

$$\sum_{i\in\mathcal{N}\setminus\{n+1\}} x_{vi(n+1)} = 1 \qquad v\in\mathcal{V} \tag{3}$$

$$\sum_{j\in\mathcal{N}} x_{vji} - \sum_{j\in\mathcal{N}} x_{vij} = 0 \qquad v\in\mathcal{V}, i\in\mathcal{N}\setminus\{0,n+1\} \tag{4}$$

$$y_{vi} - \sum_{j\in\mathcal{N}\setminus\{i\}} x_{vij} = 0 \qquad v\in\mathcal{V}, i\in\mathcal{N} \tag{5}$$

$$\sum_{v\in\mathcal{V}} y_{vi} + u_i = 1 \qquad i\in\mathcal{N}\setminus\{0,n+1\} \tag{6}$$

$$l_{vij} \leq (Q_v - P_j)x_{vij} \qquad v \in \mathcal{V}, i \in \mathcal{N}, j \in \mathcal{N}^P \tag{7}$$

$$l_{vij} \leq Q_v x_{vij} \qquad v \in \mathcal{V}, i \in \mathcal{N}, j \in \mathcal{N}^D \tag{8}$$

$$l_{vij} \geq P_i x_{vij} \qquad v \in \mathcal{V}, i \in \mathcal{N}^P, j \in \mathcal{N} \tag{9}$$

$$l_{vij} \geq D_j x_{vij} \qquad v \in \mathcal{V}, i \in \mathcal{N}, j \in \mathcal{N}^D \tag{10}$$

$$l_{vij} \geq (P_i + D_j)x_{vij} \qquad v \in \mathcal{V}, i \in \mathcal{N}^P, j \in \mathcal{N}^D \tag{11}$$

$$\sum_{i \in \mathcal{N}} l_{vij} + P_j x_{vjh} - l_{vjh} + Q_v x_{vjh} \leq Q_v \qquad v \in \mathcal{V}, j \in \mathcal{N}^P, h \in \mathcal{N} \tag{12}$$

$$\sum_{i \in \mathcal{N}} l_{vij} - D_j x_{vjh} - l_{vjh} + Q_v x_{vjh} \leq Q_v \qquad v \in \mathcal{V}, j \in \mathcal{N}^D, h \in \mathcal{N} \tag{13}$$

$$\sum_{j \in \mathcal{N}^D} D_j y_{vj} - l_{v0i} + Q_v x_{v0i} \leq Q_v \qquad v \in \mathcal{V}, i \in \mathcal{N} \tag{14}$$

$$l_{vi(n+1)} - \sum_{j \in \mathcal{N}^P} P_j y_{vj} + Q_v x_{vi(n+1)} \leq Q_v \qquad v \in \mathcal{V}, i \in \mathcal{N} \tag{15}$$

$$t^{TC} \geq \left(\sum_{i \in N} \sum_{j \in N} T_{(k+1)ij} x_{(k+1)ij} \right) \frac{1}{H} \tag{16}$$

$$T_v^{MIN} + \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} T_{vij} x_{vij} - T_v^{MAX} \leq t_v^D \qquad v \in \mathcal{V} \setminus \{k+1\} \tag{17}$$

$$0 \leq t_v^D \leq \Gamma \qquad v \in \mathcal{V} \setminus \{k+1\} \tag{18}$$

$$\sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{S}} x_{vij} \leq |\mathcal{S}| - 1 \qquad v \in \mathcal{V}, \mathcal{S} \subset \mathcal{N}, |\mathcal{S}| \geq 2 \tag{19}$$

$$x_{vij} \in \{0, 1\} \qquad v \in \mathcal{V}, i \in \mathcal{N}, j \in \mathcal{N} \tag{20}$$

$$y_{vi} \in \{0, 1\} \qquad v \in \mathcal{V}, i \in \mathcal{N} \tag{21}$$

$$u_i \in \{0, 1\} \qquad i \in \mathcal{N} \tag{22}$$

$$t^{TC} \in \mathbb{Z}^+ \tag{23}$$

The objective function (1) consists of four parts. The first term summarizes the costs related to sailing and servicing nodes for all OSVs, while the second term express the cost related to chartering an OSV from the spot market. The third and fourth term are artificial costs that penalize orders that are postponed and OSVs that return to the onshore supply depot later than planned, respectively. Constraints (2) and (3) ensure that all voyages begin and end at the depot, while constraints (4) conserve the flow through the problem defining network. The auxiliary variables are set by constraints (5), and constraints (6) ensure that all nodes are either serviced by an OSV or the visit is postponed until a later voyage. Further, constraints (7) – (11) ensure that the capacity of the OSV is not violated on any arc along the route, while constraints (12) and (13) are the cargo flow conservation constraints. Since the model does not distinguish between cargo that is to be delivered to an installation and backload, constraints (14) ensure that the total amount of cargo to be delivered to installations on a voyage equals the load on-board when the

OSV leaves the depot. Similarly, constraints (15), together with constraints (7) and (8), ensure that the load on-board when the OSV arrives at the depot equals the total amount of picked up cargo on a voyage. If an OSV is chartered from the spot market, constraint (16) calculates the time it is used, and rounds up to the nearest number of whole days. Constraints (17) calculate the delay of each OSV when returning to the depot, while constraints (18) assure that the delay is not more than Γ hours for each OSV in the long-term fleet. Finally, constraints (19) are the subtour eliminating constraints, and constraints (20) - (23) define the domain of each set of variables.

4. Variable neighborhood search heuristic

To solve the problem presented in Section 3 we propose a variable neighborhood search (VNS) heuristic with perturbations, inspired by Polat et al. [16]. Algorithm 1 shows, at an aggregate level, how the heuristic works. First, a construction heuristic is run to construct an initial solution which is used as input for the VNS heuristic (line 3). The initial solution is then run through an improvement phase that alternates between performing *shaking and local search* to improve the solution and using a *destroy* operator to obtain a new starting point, until the maximal time (maxTime) of the algorithm is reached (lines 4-13). The shaking and local search is run until no improved solution has been found for s_{max} iterations (lines 5-11). Then the best found solution thus far is destroyed to obtain a new starting point for the shaking and local search (line 12).

Algorithm 1 VNS heuristic

```
1: Input: maxTime, $s_{max}$
2: $x^k$: solution $k$
3: $x^0 = x^1 =$ constructInitialSolution() (Algorithm 2)
4: while time < maxTime do
5:     for $s = 0$ to $s_{max}$ do
6:         $x^1 =$ shakingAndLocalSearch($x^1$) (Algorithm 3)
7:         if $f(x^1) < f(x^0)$ then
8:             $x^0 \leftarrow x^1$
9:             $s \leftarrow 0$
10:        end if
11:    end for
12:    $x^1 =$ destroySolution($x^0$)
13: end while
14: Return $x^0$
```

We give detailed descriptions of the construction heuristic that produces an initial solution in Section 4.1, the shaking and local search heuristic in Section 4.2, and finally the destroy operator in Section 4.3.

4.1. Initial solution

In Algorithm 2, the pseudocode of the procedure for finding the initial solution for the VNS heuristic is presented. As in the multi-start local search heuristic presented in Brønmo et al. [4], the procedure for constructing an initial solution is based on finding several solutions and picking the best one as the initial solution. The algorithm starts by making sure all vessels have the depot pickup and delivery nodes as their start and end nodes, respectively (lines 5–8). Then, the algorithm attempts to insert all the delivery nodes (lines 10–25). This is done by selecting a random delivery node i and vessel v. The method $\mathbf{insert}(i, v, x)$ inserts node i between the pair of nodes along the route of vessel v in solution x which gives the lowest additional cost, given that the resulting route is feasible with respect to the time and capacity restrictions of the vessel (lines 10-15). Then, the delivery node's sibling, if one exists, is added provided that the resulting voyage is feasible (lines 16-20). This is done until the all delivery nodes are added, i.e. $\mathcal{N}^D = \emptyset$, or no nodes are added for m iterations.

Next, any nodes left in $\mathcal{N}^P$ are added in the same way (lines 26–36). The solution created x^j is then compared with the best solution found x^0, and if it gives an improvement, the best found solution, and the set of postponed nodes are updated (lines 37–39). Afterwards, the same procedure is repeated $j_{max}/2$ number of times, but this time nodes are only added to OSVs in the long-term fleet, not the spot vessel (line 41). This is done to help the heuristic find solutions where the spot vessel is not used, since this might be the solution with lowest cost, especially for instances with only small or no disruptions. At the end of the algorithm, the solution with lowest cost is returned.

4.2. Shaking and local search

The shaking and local search use a VNS structure by changing the neighborhood operator used to explore the solution space. The use of shaking to generate new starting points for the local search was chosen due to the similar implementation described in Polat et al. [16], which gave high quality solutions for a VRP with simultaneous pickup and delivery. The shaking decides a search direction, and the solutions obtained by the shaking are further evaluated using a best improvement local search to further improve the solution.

Since a local optimum for one neighborhood structure might not be optimal for another neighborhood structure, a larger part of the solution space is searched before a local optimum is reached than what would have been explored with a single neighborhood structure. When a local optimum is reached with respect to all neighborhoods defined, the solution is perturbed.

The implementation described in this paper employs eight local search operators - four *intra-route* operators that move nodes within a given voyage, and four *inter-route* operators that move nodes between different voyages. The local search operators employed are shown in Figure 2 and explained below. The operators described are the same as the ones described in Caseau and Laburthe [6], Brønmo et al. [4], Korsvik et al. [13], and Polat et al. [16]. Similar to what is done by Brønmo et al. [4] and Korsvik et al. [13], we include an *artificial vessel* containing all the postponed nodes in the solution, and perform the inter-route operators on the postponed nodes just as if they

Algorithm 2 constructInitialSolution()

```
1: Input: $j_{max}$, $m$
2: $x^k$: solution $k$
3: $counter = 0$
4: for $j = 1$ to $j_{max}/2$ do
5:     for $v \in \mathcal{V}$ do
6:         insert$(0, v, x^j)$
7:         insert$(n+1, v, x^j)$
8:     end for
9:     while $\mathcal{N}^D \neq \emptyset$ and $counter \leq m$ do
10:        $i =$ selectRandom$(\mathcal{N}^D)$
11:        $v =$ selectRandom$(\mathcal{V})$
12:        $\hat{x} =$ insert$(i, v, x^j)$
13:        if $\hat{x}$ is feasible then
14:            $x^j \leftarrow \hat{x}$
15:            $\mathcal{N}^D = \mathcal{N}^D \setminus \{i\}$
16:            $\hat{x} =$ insert$(sibling(i), v, x^j)$
17:            if $x'$ is feasible then
18:                $x^j \leftarrow \hat{x}$
19:                $\mathcal{N}^P = \mathcal{N}^P \setminus \{sibling(i)\}$
20:            end if
21:        else
22:            $counter = counter + 1$
23:        end if
24:    end while
25:    $counter = 0$
26:    while $\mathcal{N}^P \neq \emptyset$ and $counter \leq m$ do
27:        $i =$ selectRandom$(\mathcal{N}^P)$
28:        $v =$ selectRandom$(\mathcal{V})$
29:        $\hat{x} =$ insert$(i, v, x^j)$
30:        if $\hat{x}$ is feasible then
31:            $x^j \leftarrow \hat{x}$
32:            $\mathcal{N}^P = \mathcal{N}^P \setminus \{i\}$
33:        else
34:            $counter = counter + 1$
35:        end if
36:    end while
37:    if $f(x^j) < f(x^0)$ then
38:        $x^0 \leftarrow x^j$
39:    end if
40: end for
41: Repeat lines 4-41, with $\mathcal{V} = \mathcal{V} \setminus \{k+1\}$
42: return $x^0$
```

were part of a regular vessel's voyage.

The intra-route operators used are *2-opt*, *3-opt*, *swap* and *insert*:

- The *2-opt* move takes two nodes from a voyage, removes the arc leaving each node, reconnects the graph, and switches the direction of visit of the intermediate nodes. In Figure 2a), the arcs (1,2) and (4,5) are deleted and replaced with (1,4) and (2,5). Note that the direction of visits for nodes 2 through 4 is reversed. The operator has a complexity of $\mathcal{O}(|\mathcal{N}|^2|\mathcal{V}|)$.

- *3-opt* is similar to the 2-opt, but instead three nodes are selected, the arc leaving each node is removed, and the graph is reconnected. The intermediate nodes between two of the chosen nodes have their direction of visit reversed. In Figure 2b) the arcs (0,1), (2,3) and (5,6) are deleted and replaced with (0,5), (3,1) and (2,6). Similar to the 2-opt case, the direction of visit for nodes 3 through 5 is reversed. The operator has a complexity of $\mathcal{O}(|\mathcal{N}|^3|\mathcal{V}|)$.

- *Swap* takes two nodes in a voyage, and swaps their location in the sequence of visits. In Figure 2c), nodes 3 and 4 are swapped. The operator has a complexity of $\mathcal{O}(|\mathcal{N}|^2|\mathcal{V}|)$.

- The *insert* operator removes one node from a voyage, and inserts it in another place in the same voyage. In Figure 2d), node 4 is inserted between nodes 1 and 2. The operator has a complexity of $\mathcal{O}(|\mathcal{N}|^2|\mathcal{V}|)$.

The inter-route operators are *exchange*, *cross*, *shift*, and *replace*.

- *Exchange* takes m sequential nodes from one voyage and exchanges them with n nodes from another voyage. In Figure 2e), m is set to 2 and n to 1, and nodes 1 and 2 from the first voyage are exchanged with node 7 from the second voyage. The operator has a complexity of $\mathcal{O}(|\mathcal{N}|^4|\mathcal{V}|^2)$ in general, but is reduced to $\mathcal{O}(|\mathcal{N}|^2|\mathcal{V}|^2)$ for fixed values of m and n. In our implementation we have chosen $m = 2$ and $0 \leq n \leq 2$.

- The *cross* operator deletes the arcs between two consecutive nodes in two different voyages, and reconnects the voyages with each other. In Figure 2f), arcs (2,3) and (5,6) are deleted and replaced with arcs (2,6) and (5,3). The operator has a complexity of $\mathcal{O}(|\mathcal{N}|^2|\mathcal{V}|^2)$.

- *Shift* removes one node from a voyage, and inserts it into another voyage. In Figure 2g), node 1 is taken from the first voyage and inserted between nodes 5 and 6 in the second voyage. The operator has a complexity of $\mathcal{O}(|\mathcal{N}|^2|\mathcal{V}|^2)$.

- *Replace* takes one node from two different voyages and exchanges their locations. In Figure 2h), nodes 1 and 6 are exchanged. The operator has a complexity of $\mathcal{O}(|\mathcal{N}|^2|\mathcal{V}|^2)$.

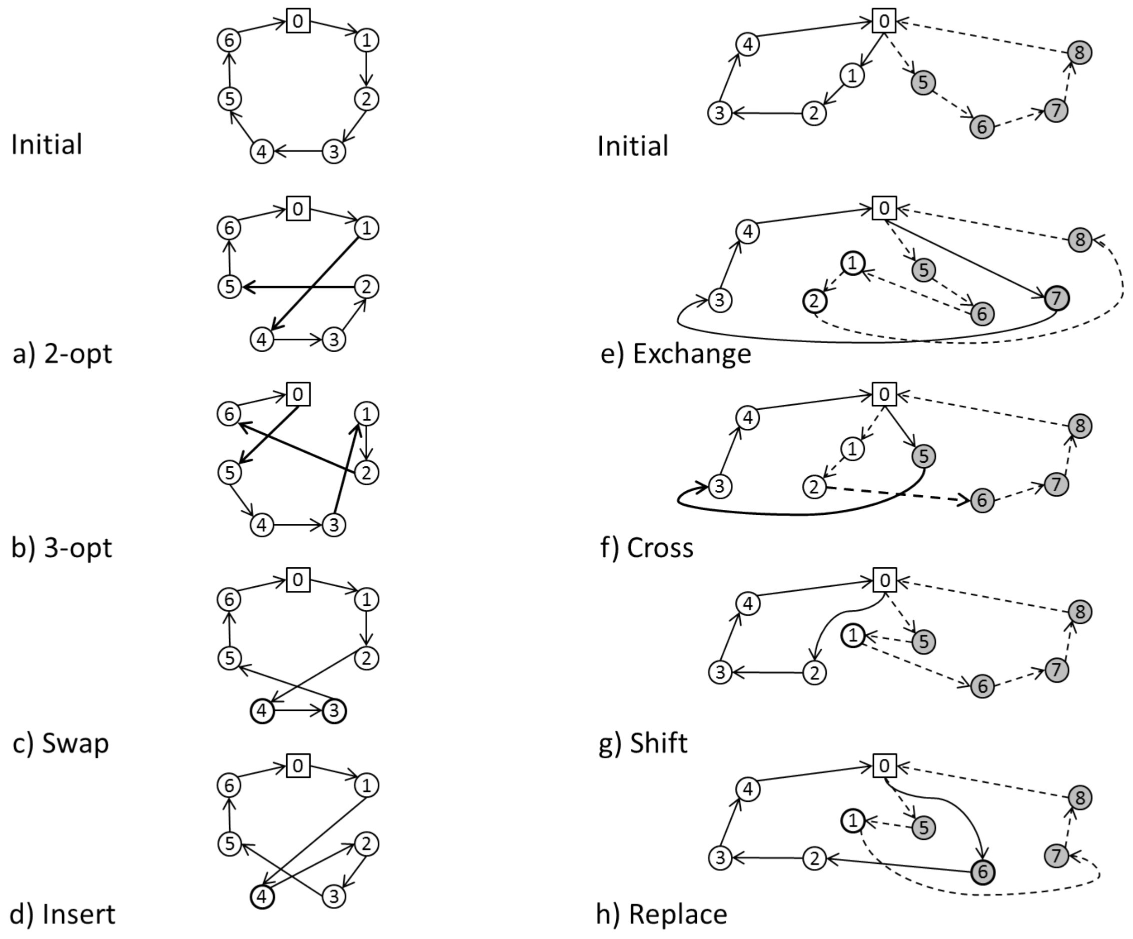

Figure 2: Local search operators used in shaking and local search. Left: intra-route operators. Right: inter-route operators.

Algorithm 3 Pseudocode for the shaking and local search

```
1: Input: $x^0$
2: $x^k$: solution $k$
3: $x^1 \leftarrow x^0$
4: for $k = 1$ to 8 do
5:     $x^2 \in N^k(x^1)$ selected randomly
6:     for $m = 1$ to 8 do
7:         $x^3 = argmin\{f(x) : x \in N^m(x^2)\}$
8:         if $f(x^3) < f(x^2)$ then
9:             $x^2 \leftarrow x^3$
10:            $m \leftarrow 0$
11:        end if
12:    end for
13:    if $f(x^2) < f(x^1)$ then
14:        $x^1 \leftarrow x^2$
15:        $k \leftarrow 0$
16:    end if
17: end for
18: Return $x^1$
```

Algorithm 3 describes the shaking and local search. An initial solution (x^0) is taken as input, and set as the best solution (x^1) (line 3). The outer loop of the heuristic controls which neighborhood is used for the *shaking* operations (lines 4-17). The shaking is performed by taking a random neighbor of x^1 in the k-th neighborhood ($N^k(x)$), producing an initial solution for the local search x^2 (line 5). The neighborhood operators ($N^k(x)$) are numbered from one to eight, e.g. if $k = 5$, the neighborhood operator numbered five is chosen. The neighborhood operators are ordered according to increasing computational complexity.

The local search iterates through all the neighborhoods, and for each neighborhood it selects the neighbor solution (x^3) with the smallest objective value (line 7). If the new solution is an improvement, it is stored and the neighborhood counter is re-set. Otherwise the local search moves on to explore the next neighborhood (line 6–12). The solution returned from the local search (x^2) is then compared with the best solution (x^1) (lines 13–16), and if it is improving, the best solution is updated, and the shaking restarts from the first operator. Otherwise, the shaking continues with its next operator. Finally, when no improving solution has been found by performing shaking and local search in any of the neighborhoods, the algorithm terminates and returns the best found solution x^1.

4.3. Destroy operator

The shaking and local search usually end up in a local optimum. To get a new starting point for the shaking and local search, the currently best known solution is destroyed in the following way. For each vessel, one randomly selected node is chosen and moved from the vessel route and to the set of postponed nodes. If the node's sibling is present in the same route, it is also moved. Note that we do not need to repair the solution again, since we treat the set of postponed nodes as an artificial vessel voyage in the local search, thus the *Shift* operator will move nodes from the list of postponed nodes and to vessel voyages, as long as it improves the objective value.

5. Computational study

In this section we present extensive computational experiments with the VNS heuristic presented in Section 4. First, the test instances are described in Section 5.1, before testing the effect of the most important parameter used in the heuristic in Section 5.2. Finally, we compare the performance of the heuristic with the results obtained by Albjerk et al. [1] in Section 5.3. The VNS heuristic is implemented in Java, and tested on a computer running Windows 7 with an Intel i7-3770 3.40 GHz CPU and 16 GB of RAM.

5.1. Test instances

The test instances are based on data supplied by Statoil, and consist of an onshore supply depot and a set of offshore installations. At the time when this study was initiated, each onshore supply depot serviced up to 13 offshore installations, which gives the corresponding instance up to 28 nodes (two nodes for each installation and two depot nodes). However, after a reorganization of the offshore supply service, one of the onshore supply depot (Mongstad) services up to around 26 offshore installations (i.e. 54 nodes). Therefore, we also test larger instances up to this size.

A summary of the test instances is given in Table 1, where the id of each instance (ID) is given together with the name of the associated depot (Depot), the number of nodes (# Nodes), and the number of OSVs (# OSVs) available. Based on input from Statoil regarding what are the most common situations, we test three variants of each of these instances:

- **No disruptions.** We create a plan for each OSV for the given instance where all vessels have normal sailing speeds and all cargoes are of (roughly) normal size. These instances are denoted using the standard IDs from Table 1.
- **Reduced sailing speed** due to adverse weather conditions. This is done by reducing the speed of each vessel from ten to five knots, and thus, increasing the sailing time. The increase in sailing time will also affect the sailing costs. The ID for these instances have a superscript "S" for speed, e.g. M_{22}^S.
- **Large cargo volumes**, which often is the case after periods where adverse weather conditions have made supplying the offshore installations impossible or too costly. This is done by setting the demand and backload volumes to the triple of what are used in the non-disrupted case to ensure that deck capacity becomes a binding constraint. The ID for these instances have a superscript "L" for load, e.g. M_{22}^L.

ID	**Depot**	**# Nodes**	**# OSVs**
M_{10}	Mongstad	10	2
M_{12}	Mongstad	12	2
Å$_{14}$	Ågotnes	14	2
F_{16}	Florø	16	2
Å$_{18}$	Ågotnes	18	3
M_{22}	Mongstad	22	3
F_{24}	Florø	24	3
Å$_{26}$	Ågotnes	26	4
M_{42}	Mongstad	42	5
M_{54}	Mongstad	54	6

Table 1: Test instances used to compare models and solution methods in Section 5.3. #OSVs includes one spot vessel.

The reason for this choice of disruptions is that both speed reduction and high cargo volumes are the most common consequences of bad weather conditions. Either the speed of each vessel is reduced for a period of time, or the cargo has piled up both at the depot and at the installations because the OSVs have been prevented from sailing for a few days. The addition of a high priority cargo does not affect the problem to a great extent, since it only modifies the number of nodes in the problem, or the size of a single cargo. Thus, the ability of the solution methods to handle a high priority cargo is sufficiently tested by the "no disruptions" variant of each instance.

For all test instances, we have assumed that 80 % of the nodes have normal priority and 20 % (randomly chosen) have high priority. The cost of postponing these are set at $200,000$ and $300,000$ NOK, repsectively. Further, we have set the delay penalty at $50,000$ nok pr. hour, and the fixed cost of chartering a spot vessel has been estimated at $275,000$ NOK pr. day.

5.2. Parameter testing for the VNS heuristic

In the VNS heuristic presented in Section 4, the s_{max} parameter controls the trade-off between shaking and local search, and perturbations done in the VNS. The higher the value of s_{max} is, the more iterations of shaking and local search are performed before a larger perturbation of the solution is done. In the following we test how the performance of the VNS heuristic changes for different values of s_{max}. The tests have been run on the three largest test instances: Å$_{26}$, M_{42} and M_{54}, and for all three variants of each instance. The reason for choosing these instances are that it is for these instances that the solutions are expected to vary the most, since they are the largest, and thus the most complicated, instances.

When testing the s_{max} parameter, the parameter is calculated as $s_{max} = \lceil s \text{ x } (N/2) \rceil$, where $s \in \{10^{-6}, 1, 3, 5, 10, 15, \infty\}$ and N is the number of nodes in the problem instance. Note that setting $s = 10^{-6}$, corresponds to having only one iteration of shaking and local search before the perturbation, while $s = \infty$ means that only shaking and local search is performed until the maximum computing time is reached. Since the heuristic is to be used in a disruption management setting, short computing time is essential. We therefore set the maximum running time to 600 seconds. Each test instance was run 10 times for each parameter setting.

The results are presented in Table 2. The table shows the best objective value found for each test instance and variant across all parameter settings, as well as the average deviation from this best known solution for each parameter setting. On average, $s = 5$ gives the lowest deviations. However, all parameter settings, except $s = \infty$ gives quite similar average deviations (between 0.58 % and 0.80 %). This indicates that the solution method is quite robust with respect to the choice of this parameter value. It is further, interesting to note that $s = 10^{-6}$ performs quite well, and is actually the best setting for Å_{26}^{L}, showing that an even simpler heuristic without shaking, also performs well on this problem.

Table 2: The best solution found for each instance over all runs with all settings and the average deviations from this best value for each value of the s parameter.

	Best obj. val.	10^{-6}	1	3	5	10	15	∞
Å_{26}	472,051	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %
Å_{26}^{S}	2,592,701	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %
Å_{26}^{L}	1,265,821	0.02 %	0.05 %	0.11 %	0.14 %	0.17 %	0.15 %	14.08 %
M_{42}	581,993	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %
M_{42}^{S}	2,651,919	0.00 %	0.00 %	0.02 %	0.00 %	0.31 %	0.72 %	1.08 %
M_{42}^{L}	1,593,228	4.93 %	4.87 %	5.24 %	4.52 %	5.58 %	5.32 %	6.04 %
M_{54}	71,4391	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %
M_{54}^{S}	3,154,631	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %	0.00 %
M_{54}^{L}	1,627,533	1.12 %	0.36 %	0.67 %	0.52 %	1.14 %	0.56 %	1.07 %
Avg. deviations		0.67 %	0.59 %	0.67 %	0.58 %	0.80 %	0.75 %	2.47 %
Avg. time to best (s)		92.2	112.1	97.9	146.8	119.0	128.6	60.7

It can be seen from Table 2 that for the base case instances (i.e. the ones with no disruptions), the VNS heuristic finds exactly the same objective value for all parameter settings for each test instance. The average deviations from best known solutions are also low for the reduced speed instances (in most cases less than 1 %). However, for the triple load instances, the average objective values vary much more between the different parameter settings and are also quite high on instance M_{42}^{L} (around 5 %). Even though we have let the VNS heuristic run for a maximum running time of 600 seconds, it can be interesting to know when the best solutions are obtained. The average time to the best solutions for each setting is shown in the bottom row of Table 2, and it can be noticed that there are no large differences in these values.

The results from Table 2 show that there is more variation in the deviations between the best found objective values and the average objective values on the disrupted instances. When the heuristic has to decide between postponing the visit to some installations to ensure that the OSV returns on time, and delaying the OSV to ensure that all installations are serviced, more solutions exist than in the non-disrupted variant of the same instance. Consequently, there is a higher chance that the heuristic does not explore all parts of the search space, and thus does not always find the best solution for these instances. When inspecting the solutions, one can see that this is due to the different number of nodes that are postponed (i.e. some solutions have one more node that is postponed compared to the best ones). This has a relatively large impact on the objective values since the (artificial) costs for postponing nodes are set quite high. Since the vessel capacity in these cases are fully utilized for both the OSVs in the regular fleet and the spot OSV, some nodes have to be postponed.

5.3. Comparison with an exact solution method

To verify that the VNS heuristic provides good results, we compare its obtained objective values and computing time with the results obtained by Albjerk et al. [1]. They presented an exact solution method, where all non-dominated voyages for each OSV are generated by a label setting dynamic programming algorithm, before solving a path-flow mixed integer programming (MIP) model using commercial optimization software for selecting one voyage for each OSV.

A summary of the computational results is provided in Table 3. The table presents the total computing time reported by Albjerk et al. [1] (Time), which is the sum of the time used to generate all voyages and the time spent solving the MIP-model. This is compared with the optimality gap (Avg. gap) and the average time until the optimal solution is found (Avg. time) of ten runs using the VNS heuristic with a max time of 60 seconds. For the instances where Albjerk et al. [1] could not find the optimal solution (shown with 'n/a' in the table), we have calculated the average gaps by comparing with the best solution found by the heuristic after 10 runs.

The results show that the average gaps for the heuristic are 0.00% for all the instances, i.e. it finds the optimal solutions in all ten runs for the instances where the optimal solution is known, and finds the same solutions in all 10 runs for those instances not solved to optimality. For all instances, the average times until the optimal solutions are found are also much lower for the VNS heuristic than for the exact solution method for all but the smallest instances, and the differences between the computational times increase as the instances become larger.

5.4. Analysis of penalty cost values - sensitivity analysis

In solving this problem in practice, it can be hard to give sensitive values to the penalty cost parameters for postponing a cargo, C_i^R, and for delaying the return of an OSV, C_v^D. The penalty cost C_v^D should be adjusted according to the negative effects of OSV v being delayed, which can vary from situation to situation depending, amongst others, on when the next voyage for the OSV is scheduled. For example, if the vessel has a long time of slack before the next planned voyage, delays are not that critical and C_v^D can be given a low value, and vice versa. Similarly, the value of C_i^R depends on the priority of the orders for node i. If an order with low priority is postponed, it has less negative effects on the operations at the offshore installation than if an order with high priority is postponed. In this section, we therefore analyze the effect on the solutions obtained from using different values for the penalty cost parameters, and we choose instances M_{42}^S and M_{54}^S for

	Albjerk et al. [1]	VNS heuristic	
ID	**Time [s]**	**Avg. gap**	**Avg. time to best [s]**
M_{10}	0	0.0 %	0
M_{10}^S	0	0.0 %	0
M_{10}^L	0	0.0 %	0
M_{12}	1	0.0 %	0
M_{12}^S	0	0.0 %	0
M_{12}^L	0	0.0 %	0
$Å_{14}$	13	0.0 %	0
$Å_{14}^S$	13	0.0 %	0
$Å_{14}^L$	1	0.0 %	0
F_{16}	529	0.0 %	0
F_{16}^S	553	0.0 %	0
F_{16}^L	101	0.0 %	0
$Å_{18}$	17,822	0.0 %	0
$Å_{18}^S$	9,727	0.0 %	0
$Å_{18}^L$	1 222	0.0 %	1
M_{22}	n/a	0.0 %	0
M_{22}^S	n/a	0.0 %	0
M_{22}^L	n/a	0.0 %	12
F_{24}	n/a	0.0 %	0
F_{24}^S	n/a	0.0 %	0
F_{24}^L	n/a	0.0 %	8

Table 3: Comparison of the path-flow MIP model and the VNS heuristic. For the path-flow MIP model the total computing times (Time) are presented and for the heuristic, the average gaps(Avg. gap) from the optimal objective values and the average computing times (Avg. time) in seconds over ten runs are presented.

this purpose. The reason for choosing these two instances is that the instances with decreased speed are the ones where both significant delays and postponement occurs. In the basic version of each instance (i.e. with no disruptions) there are no delays or postponements, and in the L instances, the capacities of the vessels, and not the duration of the voyages, are the limiting factors.

Table 4: Solutions obtained when changing the delay penalty cost

	M_{42}^S			M_{54}^S		
Δ delay cost	Total delay	# postponed	# Spot days	Total delay	# postponed	# Spot days
+50 %	0.34	4	3	2.43	6	3
+25 %	0.34	4	3	21.01	0	3
Base case	0.34	4	3	21.01	0	3
-25 %	15.64	0	3	21.01	0	3
-50 %	57.82	0	0	58.21	0	0

Table 4 shows how changing the value of the delay cost parameters C_v^D affect the solutions of M_{42}^S and M_{54}^S. The table gives the total delay of the vessel fleet in hours (Total delay), the number of postponed nodes (# postponed), and the number of days the spot vessel is chartered in for (# Spot days), when the cost parameter is adjusted between +50 % and −50 %.

For instance M_{42}^S we see that the base case solution already gives a solution with virtually no delay and four postponed nodes. Increasing the penalty cost in this case does not affect the solution. However, when we decrease the delay cost by 25 % we see that more vessel delays are preferred to be able to serve all nodes. When further reducing the penalty costs, the cost of delaying the vessels becomes so low that it is preferred to have almost 60 hours of delays in order to avoid chartering in a spot vessel.

For instance M_{54}^S the base case solution does not have any postponed nodes, but rather a delay of just over 21 hours. Adjusting the cost up and down by 25 % does not influence the solution, however, when increasing the penalty costs by 50 %, the delay is almost removed at the expense of six nodes being postponed. Reducing the cost by 50 % leads to a large increase in total delays in order to remove the expensive spot vessel from the solution.

Table 5: Solutions obtained when changing the postponement penalty costs

	M_{42}^S			M_{54}^S		
Δ postpone cost	Total delay	# postponed	# Spot days	Total delay	# postponed	# Spot days
+50 %	15.64	0	3	21.01	0	3
+25 %	15.64	0	3	21.01	0	3
Base case	0.34	4	3	21.01	0	3
-25 %	0.34	4	3	2.43	6	3
-50 %	1.88	18	0	22.46	12	0

Table 5 shows how changing the value of the postponed cost parameter C_i^R affects the solutions of instances M_{42}^S and M_{54}^S. For instance M_{42}^S we see that increasing the penalty cost ensures that all nodes are serviced, at the expense of about 15 hours delay over the entire fleet. Reducing the postpone costs by 50 % gives a solution where it is so inexpensive to postpone that 18 nodes are not serviced in order to avoid chartering in a spot vessel.

For instance M_{54}^S the trend is similar, however, here no nodes are postponed in the base case, so increasing the penalty further has no effect. For this instance, we see that a 25 % reduction in the postponed cost gives six postponed nodes but decreased delay, while a 50 % reduction leads both to increased delay and increased number of postponed nodes to avoid chartering in a spot vessel.

6. Concluding remarks

This paper studied operational planning and disruption management in offshore supply logistics, which deals with determining optimal routes for a set of offshore supply vessels servicing a number of offshore installations from an onshore supply depot. The planning problem was modeled as a pickup and delivery problem with time limits for departure and arrival at the depot, and a new variable neighborhood search (VNS) heuristic was proposed to obtain good solutions in short time for realistic problem instances. Computational tests showed that the VNS heuristic produced high quality solutions to real size problem instances with up to 54 nodes. Both significant reductions in costs and time spent on planning can be achieved from using the VNS heuristic for decision support.

[1] Albjerk, N., Danielsen, T., Krey, S., Stålhane, M., Fagerholt, K., 2016. A vessel pickup and delivery problem from the disruption management in offshore supply vessel operations. In: Paias, A., Ruthmair, M., Voß, S. (Eds.), Computational Logistics: 7th International Conference, ICCL 2016, Lisbon, Portugal, September 7-9, 2016, Proceedings. Springer International Publishing, pp. 50–64.

[2] Battarra, M., Cordeau, J.-F., Iori, M., 2014. Pickup-and-delivery problems for goods transportation. In: Toth, p., Vigo, M. (Eds.), Vehicle Routing: Problems, methods, and applications, 2nd Edition. Series on Optimization. MOS-SIAM, pp. 161–191.

[3] Berbeglia, G., Cordeau, J.-F., Gribkovskaia, I., Laporte, G., 2007. Static pickup and delivery problems: a classification scheme and survey. TOP 15 (1), 1–31.

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

[5] Brouer, B. D., Dirksen, J., Pisinger, D., Plum, C. E., Vaaben, B., 2013. The vessel schedule recovery problem (VSRP) – a MIP model for handling disruptions in liner shipping. European Journal of Operational Research 224 (2), 362 – 374.

[6] Caseau, Y., Laburthe, F., Oct. 1999. Heuristics for large constrained vehicle routing problems. Journal of Heuristics 5 (3), 281–303.

[7] Dell'Amico, M., Righini, G., Salani, M., 2006. A branch-and-price approach to the vehicle routing problem with simultaneous distribution and collection. Transportation Science 40 (2), 235–247.

[8] Gribkovskaia, I., Laporte, G., Shlopak, A., 2007. A tabu search heuristic for a routing problem arising in servicing of offshore oil and gas platforms. Journal of the Operational Research Society 59 (11), 1449–1459.

[9] Halvorsen-Weare, E., Fagerholt, K., 2011. Robust supply vessel planning. In: Pahl, J., Reiners, T., Voï¿$\frac{1}{2}$, S. (Eds.), Network Optimization. Vol. 6701 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 559–573.

[10] Halvorsen-Weare, E. E., Fagerholt, K., Nonås, L. M., Asbjørnslett, B. E., 2012. Optimal fleet composition and periodic routing of offshore supply vessels. European Journal of Operational Research 223 (2), 508 – 517.

[11] Karlaftis, M. G., Kepaptsoglou, K., Sambracos, E., 2009. Containership routing with time deadlines and simultaneous deliveries and pick-ups. Transportation Research Part E: Logistics and Transportation Review 45 (1), 210 – 221.

[12] Kjeldsen, K. H., 2012. Routing and scheduling in liner shipping. Ph.D. thesis, Aarhus University, 166 pages.

[13] Korsvik, J. E., Fagerholt, K., Laporte, G., 2009. A tabu search heuristic for ship routing and scheduling. Journal of the Operational Research Society 61 (4), 594–603.

[14] Nagy, G., Wassan, N. A., Speranza, M. G., Archetti, C., 2015. The vehicle routing problem with divisible deliveries and pickups. Transportation Science 49 (2), 271–294.

[15] Polat, O., Günther, H.-O., Kulak, O., 2014. The feeder network design problem: Application to container services in the black sea region. Maritime Economics & Logistics 16 (3), 343–369.

[16] Polat, O., Kalayci, C. B., Kulak, O., Gï¿½nther, H.-O., 2015. A perturbation based variable neighborhood search heuristic for solving the vehicle routing problem with simultaneous pickup and delivery with time limit. European Journal of Operational Research 242 (2), 369–382.

[17] Regjeringen, 2014. Facts 2014 - the Norwegian petroleum sector (in Norwegian).
URL `https://www.regjeringen.no/globalassets/upload/oed/pdf_filer_2/faktaheftet/fakta2014og/fakta`

[18] Shyshou, A., Gribkovskaia, I., Laporte, G., Fagerholt, K., 2012. A large neighbourhood search heuristic for a periodic supply vessel planning problem arising in offshore oil and gas operations. INFOR: Information Systems and Operational Research 50 (4), 195–204.

[19] Sopot, E., Gribkovskaia, I., 2014. Routing of supply vessels to with deliveries and pickups of multiple commodities. Procedia Computer Science 31 (0), 910 – 917.

[20] Statoil, 2014. Logistikkportalen.
URL `http://www.logistikkportalen.no/forsyningskjeden/behov`

[21] Statoil, 2015. 2014 statutory report.
URL `http://www.statoil.com/no/InvestorCentre/AnnualReport/AnnualReport2014/Documents/DownloadCe`