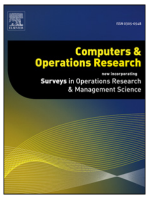
Computers and Operations Research 111 (2019) 285–296

Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/cor



Bi-objective offshore supply vessel planning with costs and persistence objectives

Thomas Borthen[a], Henrik Loennechen[a], Kjetil Fagerholt[a,*], Xin Wang[a], Thibaut Vidal[b]

[a] *Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology (NTNU), Trondheim, Norway*
[b] *Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio), Brazil*

ARTICLE INFO

Article history:
Received 11 June 2018
Revised 24 June 2019
Accepted 25 June 2019
Available online 2 July 2019

Keywords:
Multi-objective optimization
Offshore logistics
Periodic vehicle routing
Genetic algorithm

ABSTRACT

This paper introduces a bi-objective model for the offshore supply vessel planning problem (SVPP) in the oil & gas industry. The SVPP consists of determining a new weekly plan for sailing the platform supply vessels whenever the sailing plan needs revising due to some major events, such as the arrival or removal of drilling rigs, and demand increase or reduction in the current platforms. Apart from remaining cost-efficient, the new weekly plan is required to be *persistent*, i.e., exhibiting few changes from the previous plan. To achieve this, we propose a bi-objective optimization framework that enables the planners to simultaneously take into account both costs and persistence-related objectives. The framework encompasses a bi-objective SVPP model and a genetic search algorithm adapted from Borthen et al. (2018). We show that the proposed algorithm is able to provide high-quality solutions in reasonable time and has the decision support capability in real offshore operation planning.

1. Introduction

In the offshore oil and gas industry, *Platform Supply Vessels* (PSVs) provide crucial logistic support for the daily operations of offshore oil platforms (also called offshore installations). These specially designed ships transport goods, tools, equipment and personnel to and from offshore installations, and keep the installations provisioned and supplied for smooth and continuous production. The Norwegian State Oil Company, Equinor (previously called Statoil), acquires PSVs on time charter to supply their offshore installations on the Norwegian continental shelf. The PSVs are a costly resource, as the charter rates and the fuel costs for operating the PSVs are expensive. Efficient use of the PSVs is therefore of great interest.

Upon request by Equinor, Halvorsen-Weare et al. (2012) studied the *Supply Vessel Planning Problem* (SVPP) with a set of offshore installations, each requiring several services per week, and one onshore supply depot where PSVs load supply cargoes and discharge back-loads from the installations. The aim was to determine the optimal fleet of PSVs and their corresponding weekly sailing plan, i.e., the routes and schedules from the onshore depot, in order to minimize the total expenses related to chartering and operating the PSVs while fulfilling the demand for services of all installations. Fig. 1 illustrates two example weekly sailing plans for an SVPP including one PSV servicing four and five offshore installations from an onshore supply depot by sailing two and three voyages every week, respectively, in the base plan and the updated plan. Each voyage starts and ends at the supply depot (the arcs back to the depot are not shown in the figure). In the base plan, servicing four installations, the first voyage starts every Monday from the supply depot and visits installations 1, 2 and 4 prior to sailing back to the supply depot; its second voyage, on the other hand, starts every Thursday and visits installations 4, 3 and 2.

The weekly sailing routes and schedules for the PSVs are normally valid for several weeks or months until some changes are observed or expected. These may include, e.g., the arrival or removal of drilling rigs, and demand changes at the installations. When such events take place, the planners at Equinor would perform a rescheduling by solving the SVPP again with updated information. However, the new solution after rescheduling is often very different from the previous solution prior to the change. For example in Fig. 1, the new installation 5 has arrived and needs two weekly services. To accommodate this, a new updated plan with three PSV voyages is proposed, which changes the times for when some offshore installations are serviced. For example, installation 4, which was previously (in the base plan) serviced by two voy-

* Corresponding author.
E-mail address: kjetil.fagerholt@ntnu.no (K. Fagerholt).

https://doi.org/10.1016/j.cor.2019.06.014

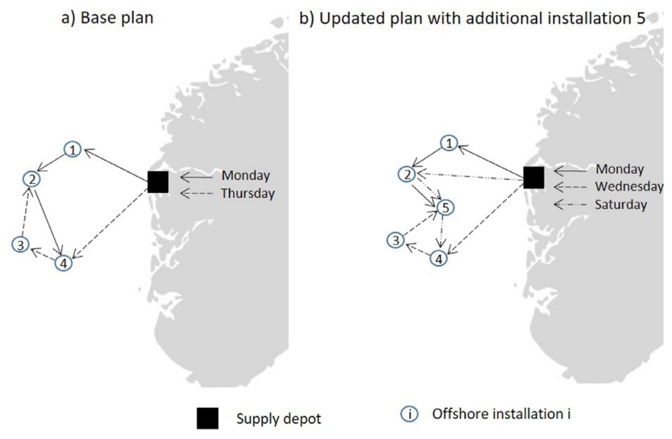


Fig. 1. Two example weekly plans for a vessel servicing four and five installations.

ages starting on Monday and Thursday, respectively, is now serviced from Wednesday and Saturday voyages according to the updated plan. This can lead to disruption to the operations at the installation, as the facilities and workforce establish their own schedules and shifts according to the weekly plan of the PSVs. When too many installations must adapt to such disruptions, the extra costs for these adaptions might no longer be negligible when planning new routes and schedules. The planners are therefore interested in new solutions (weekly plans) that are *persistent*, i.e., close to or exhibiting few changes from the current plan, the *baseline solution*.

The contribution of this paper is the development of a bi-objective framework that encompasses a bi-objective SVPP model and an efficient solution approach that can find and identify a set of high quality solutions to offshore supply vessel planning problems. We show that the Hybrid Genetic Search with Adaptive Diversity Control (HGSADC) proposed in Borthen et al. (2018) for solving the single-objective SVPP and for other routing problems in Vidal et al. (2012) can be generalized into a simple bi-objective solution approach for this problem with a limited number of adaptations and extensions: (1) generalization of the fitness functions, (2) addition of new local search operators and 3) the management of an archive of non-dominated solutions. Furthermore, we demonstrate using a set of real instances that this approach can provide valuable decision support for Equinor, regarding the tradeoffs between cost and persistence when planning for new routes and schedules for the PSVs. Moreover, we emphasize that the proposed framework can be modified and applied to other bi-objective or multi-objective contexts in offshore logistics planning where other objectives are considered.

The remainder of the paper is organized as follows. Section 2 discusses the relevant studies in the literature. The bi-objective SVPP model is proposed in Section 3. The solution method is described in Section 4, while Section 5 presents the computational study. We conclude in Section 6.

2. Literature review

There are a number of papers in the literature that consider the SVPP in offshore logistics. Common to all these studies are the determination of the periodic (weekly in most cases) routes and schedules for the voyages sailed by the PSVs, where each voyage represents a route with a start day, a duration and a visiting sequence of installations. The SVPP therefore shares many similarities with the Periodic Vehicle Routing Problem (PVRP), discussed in the survey of Francis et al. (2008).

A simplified version of the SVPP where the "periodicity" requirements on the installation visits were disregarded was first studied by Fagerholt and Lindstad (2000). These requirements were later included by Halvorsen-Weare et al. (2012) who presented a voyage-based model using pre-generated voyages for all PSVs, whereas Shyshou et al. (2012) proposed a large neighborhood search heuristic to solve the SVPP. Recently, Borthen et al. (2018) presented a genetic search-based heuristic for solving the problem, which is able to provide better solutions using shorter computation time compared with methods of Halvorsen-Weare et al. (2012) and Shyshou et al. (2012). Some studies also extend the SVPP by considering uncertain weather conditions and emissions, see for example Halvorsen-Weare and Fagerholt (2011), Norlund and Gribkovskaia (2013, 2017), Norlund et al. (2015), and Kisialiou et al. (2018a). Furthermore, Kisialiou et al. (2018b) considered flexible departure times for the PSVs and vessel coupling, where installations have a one-week planning horizon and the PSVs have a two-week planning horizon, and where PSVs can swap their schedules in the second half of their planning horizon.

In this paper, we propose a bi-objective optimization model that extends the model and solution method of Borthen et al. (2018) with the addition of a persistence-related objective. Brown et al. (1997b) pointed out that lack of persistence is a major source of complaints when optimization models are used in real life, and that new solutions that "retain the features of prior published plans" are more acceptable to decision makers than solutions that require more changes. Brown et al. (1996) described a problem of scheduling coast guard district cutters, where the original objective is replaced with a surrogate objective that preserves persistence when revising an accepted schedule. Brown et al. (1997a) optimized submarine berthing with a persistence incentive, by introducing a penalty in the objective function for moving submarines when the current berthing plan is revised, whereas Fagerholt et al. (2009) studied persistence in ship routing and scheduling. The term *consistency* is also frequently used as a synonym for *persistence* in the literature. The bi-objective SVPP shares common traits with the consistent vehicle routing problem (Kovacs et al., 2015), in which consistency is measured relatively to customer's service times and the assignment of drivers to customers. Moreover, consistent delivery quantities are sometimes desirable in inventory routing solutions (Coelho et al., 2012). In the SVPP, consistency (or persistence) is mainly relevant for service times at offshore installations. More precisely, as explained in Section 3.1, our application case requires to consider consistency in the departure days of vessels visiting the installations.

It should be noted that in this paper, we have, in order to focus on the bi-objective with the persistency, made two simplifications compared to some previous studies on the SVPP, see for example Halvorsen-Weare et al. (2012), Shyshou et al. (2012), and Kisialiou et al. (2018a). Firstly, we assume a homogeneous fleet of PSVs. Secondly, we do not consider that some offshore installations might be closed for service during nights. The reasons for making these simplifications, in addition to keeping the focus on the bi-objective model, are that the company considers all PSVs as similar for all practical purposes in the SVPP, and that the number of installations that are closed for service during nights is small.

The bi-objective SVPP that we consider in this paper is, like the consistent vehicle routing and inventory routing problems discussed above, a type of *multi-objective optimization* problem where more than one (and usually conflicting) objectives are to be optimized simultaneously. We refer to Deb (2014) for an introduction to multi-objective optimization. The optimal solutions to a multi-objective optimization problem, all having the property that one cannot improve one objective without impairing other objectives (referred to as *Pareto-optimal* or *non-dominated*), make up a so-called *Pareto front* in the objective space. The advantage of presenting a Pareto front is that decision makers can easily see how much an improvement in one objective affects the remaining objectives, and may subsequently use their preference for each objective to select which solution to use.

For multi-objective optimization problems, it is often computationally infeasible to find the entire Pareto front, and most approaches aim to find an approximation. One of the common approaches is the ϵ*-constraint method*. The main idea is to optimize only one of the objectives while adding the remaining objectives as constraints, each limited by some value referred to as ϵ. When using the ϵ-constraint method, the problem needs to be solved once for a number of ϵ values to create a Pareto front. Therefore a challenge is setting ϵ values that give enough granularity without too much computation time.

Genetic algorithms are also popular methods for solving multi-objective optimization problems. A genetic algorithm (Mitchell, 1998) is a method based on natural selection that "evolves" a population of *individuals*, representing solutions towards the optimal solution. The reason for the popularity of genetic algorithms in multi-objective optimization is that they are population-based and can thus search multiple regions of the search space simultaneously (Konak et al., 2006). In addition, they require neither convex, continuous nor unimodal (having only one maximum) solution spaces.

3. The bi-objective offshore supply vessel planning problem

In what follows, we describe the bi-objective SVPP in Section 3.1, and present its mathematical model in Section 3.2.

3.1. Problem statement

Each PSV can sail more than one *voyage* during a week, where each voyage should respect a minimum and maximum duration limit stated in days (e.g., two and three days, respectively). Each voyage should also respect a minimum and maximum limit on the number of installations visited. Indeed, short voyages lead to unexploited capacity and long ones involve too much uncertainty. Each voyage starts and ends at the supply depot. The PSVs' capacities further restrict the number and selection of installations that are visited along a voyage. Since the supply depot has limited opening hours (between 8 am and 4 pm on weekdays) and the time for preparing a PSV for a new voyage (e.g., loading/unloading) takes one working day, a PSV should arrive at the supply depot before 8 am to be able to start a new voyage on the same day. For similar reasons, only a limited number of PSVs can be prepared for a new voyage in any given day.

Each installation demands a certain number of visits during a week (service frequency), and has a given service duration. For each installation, the departures of the voyages servicing this installation must be *evenly spread* throughout the week in order to provide regular supplies, and no installation can be visited more than once a day. Indeed, consider an installation that requires two services a week, and a solution in which the PSVs servicing it departs from the depot on Tuesdays and Wednesdays. If the installation places a delivery request just after the second PSV has left on Wednesday, the next departure occurs six days later, leading to very significant delays or expensive express deliveries by helicopters.

Satisfying these requirements and constraints, the weekly plan determines: for each PSV, which day to start the voyages; then for each such voyage, which installations to visit in what sequence. At any point in time, the weekly plan by which the PSVs operate is referred to as the *baseline solution*. From time to time, the existing plan or baseline solution needs to be revised. This can, for example, be the case when some offshore installations experience major demand changes (e.g., when moving from drilling to production phase), new installations are to be serviced or installations are removed and do not need service anymore, or when the number of visits at some installations changes.

When the weekly plan is to be revised, it is required that the new plan, while remaining cost-efficient, is such that the workers and managers at any installation will not experience too many *changes* in the departures of the voyages servicing the installation. For one installation, for example, a change is incurred if a previous Monday voyage servicing this installation is now serviced through a Wednesday voyage instead. A new weekly plan having few such changes is regarded as *persistent*. Therefore, whenever the baseline solution needs to be revised, the bi-objective SVPP that we address in this paper seeks to find a new weekly plan that minimizes the total costs, computed as the costs for sailing the PSVs, and at the same time minimizes its total changes from the baseline solution.

Fig. 2 illustrates three weekly plans for a small example with only one PSV and four installations, including the current plan and two possible new plans. The schedules of the voyages sailed by the PSV are indicated with rectangle boxes, and the sailing route for each voyage is represented by a sequence of installation numbers indicating the order in which the installations are visited. In this example, New Plan 1 and New Plan 2 have equal total costs because the voyages included are the same except for their starting days. However, considering the Current Plan as the baseline solution, New Plan 1 is preferable since it is more persistent than New Plan 2 from the perspective of the installations. Persistent plans make the operations at the installations more predictable for both managers and workers, who can focus on working as efficiently as possible instead of adapting to changes. For New Plan 2, the voyage servicing installation 4 is moved from Thursday in Current Plan to Monday in New Plan 2 – this leads to *two changes* for installation 4 if New Plan 2 is to replace the Current Plan (one cancellation and one addition). In fact, in this example, the number of changes for New Plan 1 from the baseline solution are {0, 0, 0, 0} for installations 1, 2, 3 and 4; whereas for New Plan 2 the number of changes are {2, 0, 2, 2}.

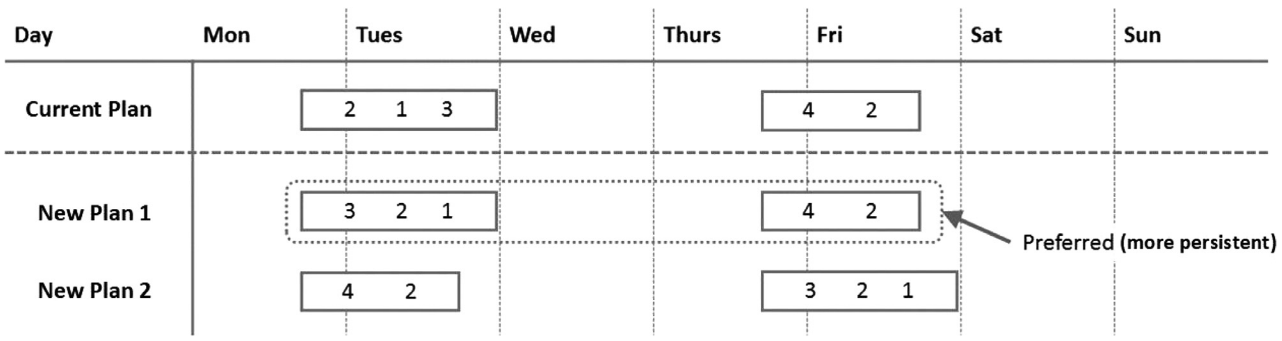


Fig. 2. Example of the current and two new weekly plans. New Plan 1 is preferred because it is more persistent.

It should be noted that we measure the solution persistence for installations relatively to the departure days of the vessels visiting them. This is a direct requirement of the company, which considers persistence in the departure days for the offshore installations much more important than persistence in the visiting days. Irregular departure days would also force the offshore installations to dynamically change their ordering procedures. Finally, we make a small simplification of the SVPP in comparison to Halvorsen-Weare et al. (2012) and Shyshou et al. (2012) by removing time-window requirements for offshore installations. This simplification has been made in accordance with the company since very few installations have such opening hours in practice (only four out of 27), such that the solutions are likely to be directly feasible or easily adjustable, if needed.

3.2. The bi-objective model

We now present the mathematical formulation of the bi-objective model for the SVPP. This model is based on Halvorsen-Weare et al. (2012) and Shyshou et al. (2012). However, it differs from previous models in that: (a) a persistence-related objective is added, and (b) the fleet of PSVs is given as input (though Section 4.3 explains how to determine the optimal fleet size whenever needed).

We use the label-setting dynamic programming algorithm proposed in Borthen et al. (2018) to generate the set of non-dominated candidate voyages for the PSVs. Furthermore, $\mathcal{V}$ is the set of PSVs, $\mathcal{N}$ the set of all offshore installations, $\mathcal{T}$ the set of days in the planning horizon, $\mathcal{L}$ the set of all possible voyage duration (in days), and $\mathcal{F}$ the set of all possible visit frequencies (number of weekly visits) for installations. Let $\mathcal{N}_f \subseteq \mathcal{N}$ be the set of installations with frequency f, $\mathcal{R}_v \subseteq \mathcal{R}$ the set of candidate voyages that PSV v can sail, $\mathcal{R}_{vi} \subseteq \mathcal{R}$ the set of candidate voyages for PSV v that visit installation i, and $\mathcal{R}_{vl} \subseteq \mathcal{R}$ the set of candidate voyages for PSV v that have duration l (in days).

We denote by C^S_{vr} the cost of using PSV v for sailing voyage r. Let S_i be the service frequency required (minimum number of weekly visits) by installation i, F_v the number of days PSV v is available to sail during the planning horizon, and B_t the upper bound on the number of PSVs that may depart the supply depot on day t. To ensure evenly spread departures we define $0 \le h_f \le |\mathcal{T}|$ to represent the length of an auxiliary sub-horizon for those installations requiring f visits per week (Shyshou et al., 2012). During any sub-horizon h_f, there must be at least $\underline{P_f}$ and no more than $\overline{P_f}$ departures to an installation whose visit frequency is f. For example an installation with $f = 3$, i.e., requiring three visits per week, would need at least one and no more than two departures every three days to ensure even spread of departures. This can be achieved by setting $h_3 = 3,\ \underline{P_f} = 1$ and $\overline{P_f} = 2$.

Finally, we define x_{vrt} to be a binary decision variable which equals one if PSV v sails voyage r starting on day t, and zero otherwise.

Based on the above definitions, we first recall the constraints of the SVPP model (Borthen et al., 2018) before discussing the two objectives:

$$\sum_{v\in\mathcal{V}}\sum_{r\in\mathcal{R}_{vi}}\sum_{t\in\mathcal{T}} x_{vrt} \ge S_i \qquad i \in \mathcal{N} \tag{1}$$

$$\sum_{l\in\mathcal{L}}\sum_{r\in\mathcal{R}_{vl}}\sum_{t\in\mathcal{T}} l x_{vrt} \le F_v \qquad v \in \mathcal{V} \tag{2}$$

$$\sum_{v\in\mathcal{V}}\sum_{r\in\mathcal{R}_{v}} x_{vrt} \le B_t \qquad t \in \mathcal{T} \tag{3}$$

$$\sum_{r\in\mathcal{R}_{vl}} x_{vrt} + \sum_{r\in\mathcal{R}_{v}}\sum_{\tau=1}^{l-1} x_{vr,(t+\tau)mod|\mathcal{T}|} \le 1 \qquad v \in \mathcal{V}, t\in\mathcal{T}, l\in\mathcal{L} \tag{4}$$

$$\underline{P_f} \le \sum_{v\in\mathcal{V}}\sum_{r\in\mathcal{R}_{vi}}\sum_{h=0}^{h_f-1} x_{vr,(t+h)mod|\mathcal{T}|} \le \overline{P_f} \qquad f\in\mathcal{F}, i\in\mathcal{N}_f, t\in\mathcal{T} \tag{5}$$

$$x_{vrt} \in \{0,1\} \qquad v\in\mathcal{V}, r\in\mathcal{R}_v, t\in\mathcal{T}. \tag{6}$$

Constraints (1) ensure that the required service frequency for each installation is satisfied. Constraints (2) ensure that each PSV does not sail more than its availability. Constraints (3) restrict the number of PSVs departing the supply depot on every given day, where *a mod b* denotes the remainder when dividing $a \in \mathbb{N}$ by $b \in \mathbb{N}$. Constraints (4) state that a PSV cannot begin a new voyage before returning from its previous one. Constraints (5) make sure that the departures to each installation are properly spread. Finally, constraints (6) enforce binary requirements on the decision variables.

3.2.1. Objective 1: minimizing cost

$$\text{minimize} \quad \sum_{v\in\mathcal{V}}\sum_{r\in\mathcal{R}_v}\sum_{t\in\mathcal{T}} C^S_{vr} x_{vrt} \tag{7}$$

The first objective function (7), minimizes the sum of all sailing costs.

3.2.2. Objective 2: minimizing changes from the baseline solution

The second objective of the model is to minimize the total number of changes in the departures of the voyages as experienced by the installations, i.e., maximizing persistence. Furthermore, we only consider the common installations that are serviced both before and after the revision of the baseline solution, i.e., we do not consider the installations that will be added or removed.

Let $\mathcal{N}^B$ be the set of installations considered in the baseline solution. We then define σ_{it} to be a binary variable that equals one if there is a PSV departure servicing installation i on day t, and zero otherwise. Recall that no installation can have more than one departure per day; σ_{it} can therefore be represented as:

$$\sigma_{it} = \sum_{v\in\mathcal{V}}\sum_{r\in\mathcal{R}_{vi}} x_{vrt}, \quad i\in\mathcal{N}\cap\mathcal{N}^B, t\in\mathcal{T}. \tag{8}$$

Then, using σ^B_{it} to represent the corresponding values as input from the baseline solution, the second objective (minimizing the total number of changes in PSV departures experienced by the installations) can be written as:

$$\text{minimize} \quad \sum_{i\in\mathcal{N}\cap\mathcal{N}^B}\sum_{t\in\mathcal{T}} |\sigma_{it} - \sigma^B_{it}|. \tag{9}$$

Objective function (9) can be linearized by introducing a binary variable γ_{it} that equals one if there is a change in departure to installation i on day t, and zero if otherwise, together with the following objective function (10) and the additional constraints (11) to (13):

$$\text{minimize} \quad \sum_{i\in\mathcal{N}\cap\mathcal{N}^B}\sum_{t\in\mathcal{T}} \gamma_{it} \tag{10}$$

$$\gamma_{it} \ge \sigma_{it} - \sigma^B_{it} \qquad i\in\mathcal{N}\cap\mathcal{N}^B, t\in\mathcal{T} \tag{11}$$

$$\gamma_{it} \ge \sigma^B_{it} - \sigma_{it} \qquad i\in\mathcal{N}\cap\mathcal{N}^B, t\in\mathcal{T} \tag{12}$$

$$\gamma_{it} \in \{0,1\} \qquad i\in\mathcal{N}\cap\mathcal{N}^B, t\in\mathcal{T}. \tag{13}$$

Algorithm 1 The HGSADC algorithm for bi-objective SVPP.

```
1: Initialize population                                                ▷ Borthen et al. (2018)
2: while Iterations without improving Pareto front $< I^{NI}$ and time $< T^{MAX}$ do
3:     Select parent individuals $s_1$ and $s_2$                          ▷ Borthen et al. (2018)
4:     Generate offspring $s_{new}$ from $s_1$ and $s_2$ (crossover)       ▷ Borthen et al. (2018)
5:     Educate offspring $s_{new}$                                         ▷ Section 4.2*
6:     if $s_{new}$ is infeasible then
7:         Repair $s_{new}$ with probability $\rho^{REP}$                  ▷ Borthen et al. (2018)
8:     end if
9:     if $s_{new}$ is still infeasible then
10:        Insert $s_{new}$ into infeasible subpopulation
11:    else
12:        Insert $s_{new}$ into feasible subpopulation
13:    end if
14:    if maximum subpopulation size $\mu + \lambda$ reached then
15:        Select survivors                                                ▷ Section 4.2*
16:    end if
17:    Penalty parameters adjustment                                       ▷ Borthen et al. (2018)
18:    if Pareto front is not improved for $I^{DIV}$ iterations then
19:        Diversify population                                            ▷ Borthen et al. (2018)
20:    end if
21:    Update the current Pareto front $\mathcal{S}^{PARETO}$              ▷ Section 4.2*
22: end while
```

4. Solution method

In this section, we present a Hybrid Genetic Search with Adaptive Diversity Control (HGSADC) algorithm for solving the bi-objective SVPP. The algorithm is adapted from the genetic method of Borthen et al. (2018) for the single-objective SVPP, which was again adapted from the HGSADC metaheuristic introduced in Vidal et al. (2012). In the present case, solving a multi-objective optimization problem amounts to producing a collection of non-dominated, possibly Pareto-optimal, solutions instead of a single one. We therefore extended the algorithm presented in Borthen et al. (2018) with a (1) generalized fitness function, (2) new local search operators to promote persistence, and (3) a management of the best non-dominated solutions in an archive. Section 4.1 provides an overview of the algorithm and Section 4.2 details the needed adaptations.

4.1. Overview of the algorithm

The general structure of our HGSADC algorithm is summarized in Algorithm 1. Through reproduction and variation, the algorithm gradually evolves an initial population of individuals to obtain better non-dominated individuals that provide an accurate approximation of the Pareto front. Algorithm 1 contains links to the sections where each method component is discussed. A star (*) sign indicates that major modifications have been made when adapting the single-objective method of Borthen et al. (2018) to our bi-objective problem. These changes are detailed in Section 4.2. The other parts of the HGSADC algorithm are only briefly summarized here, and we refer to Borthen et al. (2018) for more details.

The initialization of the population is carried out by systematically creating a predefined number of individuals and allocating them to the feasible or infeasible sub-population after performing a local search-based education procedure. Then, when generating offspring individual, each parent is selected by a binary tournament, i.e., randomly picking two individuals from the entire population and choosing the one with the best *biased fitness* (Section 4.2) as the parent. The crossover operator then takes over and combines the chromosomes of the two parent individuals into a new offspring individual. The algorithm keeps generating offspring individuals until there have been I^{NI} iterations without improvement to the Pareto front, or the maximum running time limit T^{MAX} is attained. Here, one *iteration* refers to the creation and improvement of one individual, and the Pareto front is comprised of the best non-dominated individuals found, which are stored in an additional solution archive through the search.

Each newly generated individual, either in the population initialization or through crossover, undergoes *education*, and also *repair* if necessary, to improve its performance on one or both of the two objectives. The new education procedures compared to Borthen et al. (2018) are described in Section 4.2. Throughout the search the HGSADC algorithm keeps the population separated in two disjoint subpopulations: a subpopulation of feasible individuals, and a subpopulation of infeasible individuals. Infeasible individuals are kept during the search because of the numerous observations in the literature (e.g., Cordeau et al., 2001; Vidal et al., 2014; 2015) that underline the fact that allowing a controlled exploration of infeasible solutions in the search process often improves its performance, as optimal solutions often lie at the boundary of feasibility. Here, individuals can be infeasible with respect to the constraints on voyage duration, capacity and number of visited installations.

The size of each subpopulation is governed by parameters μ and λ, where μ is the *minimum* size and λ is the *generation* size (number of offspring), such that the *maximum* subpopulation size is $\mu + \lambda$. When the maximum size of any subpopulation is reached, its individuals are removed using a *survivor selection* process until there are only μ individuals left in the subpopulation again. A *diversification* procedure is also used to recover population diversity when no improvement to the Pareto front has been made during the last I^{DIV} iterations.

4.2. Adaptations of the HGSADC algorithm

We now describe the modifications operated on the algorithm of Borthen et al. (2018) to handle the bi-objective problem.

4.2.1. Generalized fitness function

Appropriate individual fitness evaluations are essential in keeping the search guided according to the desired objectives in the survival selection process. The fitness evaluation for an individual consists in Borthen et al. (2018) of two terms. The first is the penalized cost, which is a modified cost value that takes penalties for infeasibility into account. The second term is a diversity contribution, which is included to represent the individual's representation to the population diversity. However, in our bi-objective SVPP, the second persistence objective is to minimize changes from the baseline solution. Therefore, we add a third term to the fitness evaluation: the number of changes from the baseline solution.

For a given individual s, the number of changes can easily be calculated using (8) and (9). To evaluate the overall fitness of an individual, the three measures, i.e., *penalized costs, number of changes from the baseline* and *diversity contribution*, must be considered as a whole. However, since they have different scales and units, we use *ranking* to avoid any scaling issue (Vidal et al., 2012): let $Rank^C(s)$, $Rank^P(s)$ and $Rank^D(s)$ be the ranks of individual s in terms of penalized costs, number of changes from the baseline and diversity contribution, respectively (rank 1 being the best). Then, we define the *biased fitness* of an individual as a weighted sum of all three ranks as follows:

$$BF(s) = Rank^C(s) + Rank^P(s) + \left(1 - \frac{n^{ELI}}{|\mathcal{S}|}\right) Rank^D(s), \tag{14}$$

where parameter n^{ELI} governs the weight of the diversity contribution in the evaluation. A higher value of n^{ELI} gives lower weight to diversity, meaning that the biased fitness of an individual depends mostly on the objective values of the individual.

4.2.2. New education operators

Each newly generated individual, either during initialization or through crossover of two parents, undergoes a local search-based *education* process, and also *repair* if it is infeasible, to improve its performance on one or both of the two objectives. The education process is selected among the following four procedures with equal probability:

- Cost education
- Persistence education
- First cost education, then persistence education
- First persistence education, then cost education

The cost education aims to improve the individual in terms of its performance on penalized costs through various types of local searches, e.g., changing the order of visits in each voyage, changing installation and PSV patterns, and attempting to merge two voyages into one. The persistence education, on the other hand, aims to improve the individual in terms of its number of changes from the baseline solution. We now describe in more details the persistence education; and we refer to Borthen et al. (2018) for the cost education.

Two procedures are designed for persistence-education, *changing installation pattern* and *moving PSV departures to other days*, both aiming to reduce the number of changes from the baseline solution. For each persistence education, the *changing installation pattern* procedure is applied once, followed by one run of the *moving PSV departures to other days* procedure.

Changing installation pattern. This procedure is described in Algorithm 2. It selects a random installation and changes its installation pattern to the one given by the baseline solution. The pattern for installation i of individual s is given by $\pi_i(s)$, which provides the departure days for the voyages servicing installation i. For example, $\pi_i(s) = 1, 4$ means that installation i is serviced on voyages starting on days 1 and 4. Similarly, π_i^B gives the installation patterns in the baseline solution. If the selected installation is not in the baseline solution, or the patterns are the same, a new random installation is selected. Let $\psi(i, v, t)$ be the minimum penalized cost for the insertion of installation i into the voyage sailed by PSV v on day t. Then for each day in the new pattern, the installation visit is inserted into the position that results in the lowest penalized costs.

Algorithm 2 Changing installation pattern.

```
1:  Given individual $s$
2:  while $\mathcal{N} \neq \emptyset$ do
3:      $i \leftarrow$ installation randomly selected from $\mathcal{N}$
4:      if $i \notin \mathcal{N}^B$  or  $\pi_i(s) = \pi_i^B$ then
5:          Remove $i$ from $\mathcal{N}$
6:      else
7:          Remove all visits to installation $i$ in individual $s$
8:          $\pi_i(s) \leftarrow \pi_i^B$
9:          for $t \in \pi_i^B$ do
10:             $v = \text{argmin}_{v \in \mathcal{V}}\ \psi(i, v, t)$
11:             Insert $i$ into $r_{vt}$ at the least cost position
12:         end for
13:         Return the updated individual $s$
14:     end if
15: end while
```

Moving PSV departures to other days. Similar to *changing installation pattern*, this procedure aims to transform an individual to become more similar to the baseline solution, by moving a PSV departure from one day to another and the installation visits along with it. Meanwhile it also tries to ensure that there are enough PSVs departing on each day to maintain feasibility, i.e., to service all the installations requiring a departure on that day as specified by the installation patterns.

The procedure is described in Algorithm 3 and consists of three steps: (1) selecting a day t^- to remove a PSV departure and a day t^+ to add a new PSV departure, (2) moving installation visits from day t^- to the new voyage on t^+ and (3) distributing the remaining installation visits on t^- to the remaining voyages.

In Step 1, the procedure begins by calculating the number of PSVs that need to depart on each day in order to attain the baseline solution and the number of PSVs departing in the current individual. It then selects a random pair of days (t^-, t^+), where day t^- has more PSVs departures than required to attain zero change from the baseline solution and day t^+ has fewer. Line checks that there is at least one PSV departure on t^- that can be moved to t^+, i.e., that the previous voyage of the PSV is finished before t^+. It also checks that there is sufficient depot capacity for another PSV to depart on t^+. If no such PSV exists, a new pair of days is selected. The PSV with the most time from t^+ until its next departure is chosen as the PSV v to move, since it can sail the longest voyage, and a new voyage r_{v,t^+} is created for PSV v on day t^+. In Step 2, randomly selected installations are removed from the set of installations with a departure on t^-, $\mathcal{N}_{t^-}$, and added to the end of r_{v,t^+}. This is done until the new voyage visits the maximum number of installations, or there are no more installation visits left on t^-. An installation i is moved only if it has no departure on t^+ already and if t^- is not in the baseline solution for i. Finally, the remaining installations from the removed voyage are inserted into the least cost positions in the other voyages on that day.

After the education process, if the resulting individual is feasible, it is referred to as *naturally feasible*; otherwise, *repair* takes place with a given probability. The repair procedure consists in repeating the (same type of) education process with $10\times$ higher penalty parameters. If the individual is still infeasible, the education process is run again with $100\times$ higher parameter values.

4.2.3. Management of the non-dominated solution archive

Any individual s created by the education procedure is inserted into the adequate (feasible or infeasible) subpopulation in

Algorithm 3 Moving PSV departures to other days.

```
1: Given individual $s$

   STEP 1: SELECT DAYS TO MOVE DEPARTURE FROM AND TO
2: for $t \in \mathcal{T}$ do
3:     $n_t^B \leftarrow$ Number of PSV departures required on day $t$ to attain baseline solution
4:     $n_t(s) \leftarrow$ Number of PSV departures on day $t$ in current individual $s$
5: end for
6: $\mathcal{T}^- \leftarrow \{t \in \mathcal{T} \mid n_t(s) < n_t^B\}$        ▷ Days with too few PSVs departing
7: $\mathcal{T}^+ \leftarrow \{t \in \mathcal{T} \mid n_t(s) > n_t^B\}$        ▷ Days with excess PSVs departing
8: $\mathcal{T}^C \leftarrow \mathcal{T}^- \times \mathcal{T}^+$
9: $(t^-, t^+) \leftarrow$ pair of days randomly selected from $\mathcal{T}^C$
10: if not feasible to move PSV departure from $t^-$ to $t^+$ then
11:     $\mathcal{T}^C \leftarrow \mathcal{T}^C \setminus \{(t^-, t^+)\}$
12:     if $\mathcal{T}^C \neq \emptyset$ then
13:         go to 9
14:     else
15:         Terminate procedure
16:     end if
17: end if
18: $v \leftarrow$ PSV with highest number of days from $t^+$ until next departure

   STEP 2: MOVE INSTALLATION VISITS TO NEW DAY
19: Create new voyage $r_{v,t^+}$
20: $\mathcal{N}_{t^-} \leftarrow$ installations with departure on $t^-$
21: while $|r_{v,t^+}| < N^{MAX}$ and $\mathcal{N}_{t^-} \neq \emptyset$ do
22:     $i \leftarrow$ installation randomly selected from $\mathcal{N}_{t^-}$
23:     if $i$ has no departure on $t^+$ already and $t \notin \pi_i^b$ then
24:         Remove $i$ from its voyage on $t^-$
25:         Add $i$ at the end of $r_{v,t^+}$
26:     end if
27:     Remove $i$ from $\mathcal{N}_{t^-}$
28: end while

   STEP 3: DISTRIBUTE REMAINING INSTALLATION VISITS
29: if $\mathcal{N}_{t^-} \neq \emptyset$ then
30:     for $i \in \mathcal{N}_{t^-}$ do        ▷ Remaining installations in the removed voyage, $r_{v,t^-}$
31:         $v = \text{argmin}_{v \in \mathcal{V}_{t^-}} \ \psi(i, v, t^-)$
32:         Insert $i$ into $r_{v,t^-}$ at least cost position
33:     end for
34: end if

35: Return the updated individual $s$
```

relation to its feasibility. In addition to the population, an archive $\mathcal{S}^{PARETO}$ containing the best non-dominated individuals found so far is maintained. Each individual s is compared to the set of individuals in the archive. If s is non-dominated, then it is added to $\mathcal{S}^{PARETO}$ and any individual that it dominates is excluded. Otherwise s is not added to $\mathcal{S}^{PARETO}$. The archive is not limited in size so as to contain all relevant solutions, in contrast to the population which is managed to retain a diverse and limited set of individuals by operating survivor selection phases based on their biased fitnesses (Borthen et al., 2018).

4.3. Fleet-size optimization

The algorithm described in the previous subsections considers a fixed fleet size. However, some scenario changes (e.g., the addition of services to offshore installations) can lead to infeasible problems if the fleet size remains unchanged. This section explains how to determine the fleet size in such settings. As in our computational study in Section 5, we consider here the case of a homogeneous fleet since the PSVs used by Equinor are for most practical planning purposes all similar to each other. In our application setting, finding the optimal fleet size reduces to finding the minimum feasible fleet size. Indeed, any increase of the fleet size beyond its minimum size is very costly, since the time charter cost is by far the dominant part of the total cost. Equinor would therefore never consider a solution that uses more vessels than really required, no matter how much this would increase the solution persistence.

To find this minimum fleet size, we start with a large fleet size and perform a feasibility check (i.e., run the HGSADC algorithm). As soon as one feasible solution is found, we remove one PSV from the fleet and perform another feasibility check. The process iterates until it is no longer possible to recover feasibility after the removal of a PSV, in which case the last vessel is added back to obtain the final fleet size.

It could be mentioned in the end that since they have already an existing fleet of PSVs on longer or shorter contracts (used in the baseline solution), the fleet size problem in practice usually reduces to evaluating whether the existing fleet still is sufficient, or alternatively add or remove one PSV to find a feasible solution with minimum fleet size.

Table 1
The test cases.

Test cases	Problem size		Baseline		Remarks
	# inst.	# visits	# inst.	# visits	
Case S-1	12	40	9	29	adding 3 installations
Case S-2	12	40	12	40	all load reduced by 25%
Case S-3	12	40	15	52	removing 3 installations
Case S-4	11	39	11	36	adding 3 weekly visits
Case L-1	30	87	27	80	adding 3 installations
Case L-2	27	80	27	80	all load reduced by 25%
Case L-3	22	71	27	80	removing 5 installations
Case L-4	27	83	27	80	adding 3 weekly visits
Case L-5	27	85	27	80	adding 5 weekly visits
Case L-6	29	85	27	80	adding 2 installations

5. Computational study

In this section we present a computational study using data provided by Equinor. To evaluate the performance of the proposed HGSADC algorithm, ten problem cases have been used: four small and six large (real-size) ones. For the four small cases, which have up to 12 installations, we can compare the results of the HGSADC algorithm with those of using the ϵ-constraint method on Model (1–8,10–13).

The HGSADC algorithm has been implemented in Java 1.8 and run on a MacBook Prowith an Intel Core i7-8650U CPU @ 1.90GHz, 2112Mhz, 4 Cores, 8 Logical Processors and 16GB of RAM. The ϵ-constraint method was implemented and solved using the commercial solver FICO Xpress 7.9.0 and run on a computer with a 3.4GHz Intel Core i7 processor and 16GB of RAM running Windows 7 Enterprise. The parameters in the HGSADC algorithm have been set to the same values as in Borthen et al. (2018), where it was also shown that the HGSADC algorithm produces better solutions to the single-objective version of the SVPP than previous algorithms.

In the following, we first describe the test cases in Section 5.1. Section 5.2 presents the comparison between the HGSADC algorithm and the ϵ-constraint method when solving the four small cases. Finally, the results on the six large real-size cases using the HGSADC algorithm are discussed in Section 5.3.

5.1. Test cases

All test cases consider one onshore supply depot, a planning horizon of one week and a fixed fleet of two or four identical PSVs (for small or large cases, respectively). The size of the cases varies, considering 11 to 30 offshore installations, and 39 to 87 total weekly visits. The number of weekly visits required by an installation ranges from one to five, and the service time at each installation ranges from one and a half to four hours. The supply depot is open for eight hours (08.00–16.00) from Monday to Saturday and is closed on Sunday. Also, a PSV needs to be at the supply depot before 08.00 to start on a new voyage the same day, and all voyages depart from the depot at 16.00. Finally, every voyage is constrained to visit between one and eight installations, and can last a maximum of three days, with no requirement on the minimum duration.

The test cases we use in the computational study represent typical scenarios where the weekly plan must be revised due to some major event, and are all based on the experience of the planners at Equinor. We first describe the four *small* problem cases:

- **Case S-1: Adding new installations**
The original weekly plan servicing nine installations (29 total weekly visits), which is used as baseline solution, must be revised because of the addition of three new installations. The new SVPP problem considers 12 installations and 40 weekly visits.
- **Case S-2: Load reduction**
The original weekly plan servicing 12 installations (40 weekly visits) must be revised because the demand, and hence the service time, at each installation are reduced by 25%. The original 12–40 plan before load reduction is used as baseline solution.
- **Case S-3: Shutdown of installations**
The original weekly plan servicing 15 installations (52 weekly visits) must be revised because three installations have been shut down and removed from the problem. The new SVPP problem considers 12 installations (40 weekly visits), and the original 15–52 plan is used as baseline solution.
- **Case S-4: Adding visits to installations**
The original weekly plan servicing 11 installations (36 weekly visits) must be revised because three installations now each requires one extra visit every week. The new SVPP problem considers 11 installations (39 weekly visits), and the original 11–36 plan is used as baseline solution.

In addition to the small cases, we also use six *large* real-size cases with increased number of installations and weekly visits, covering the same four types of scenarios as described above. We summarize all test cases in Table 1, where we under “Baseline” show the characteristics of the original problem for each case from which the “baseline solution” is derived.

5.2. Comparison with ϵ-constraint method on small cases

The small cases with 11 or 12 installations can be solved to optimality using the ϵ-constraint method. To implement the ϵ-constraint method to solve the bi-objective SVPP problem, we start by setting ϵ to the largest possible number of changes from the baseline solution, i.e., the maximum feasible value the objective function (10) can take, which equals $|\mathcal{N} \cap \mathcal{N}^B| \times |\mathcal{T}|$. The objective function (10) is then replaced by the following constraint:

$$\sum_{i \in \mathcal{N} \cap \mathcal{N}^B} \sum_{t \in \mathcal{T}} \gamma_{it} \leq \epsilon. \tag{15}$$

The resulting single-objective problem is solved to optimality using FICO Xpress to obtain a Pareto-optimal solution. This process is then repeated after reducing ϵ by one, until no feasible solution can be found. The final set of Pareto-optimal solutions constitutes the *optimal* Pareto front. It should be noted that this method requires pre-generating the feasible PSV voyages, which we do using the label-setting algorithm of Borthen et al. (2018).

Fig. 3 shows the results obtained by our bi-objective HGSADC algorithm for the small cases. Each case is solved ten times with the HGASDC algorithm using different initial seeds. For each case, we display the combined Pareto front obtained among all ten runs of the HGSADC algorithm, shown by the blue points. The red points

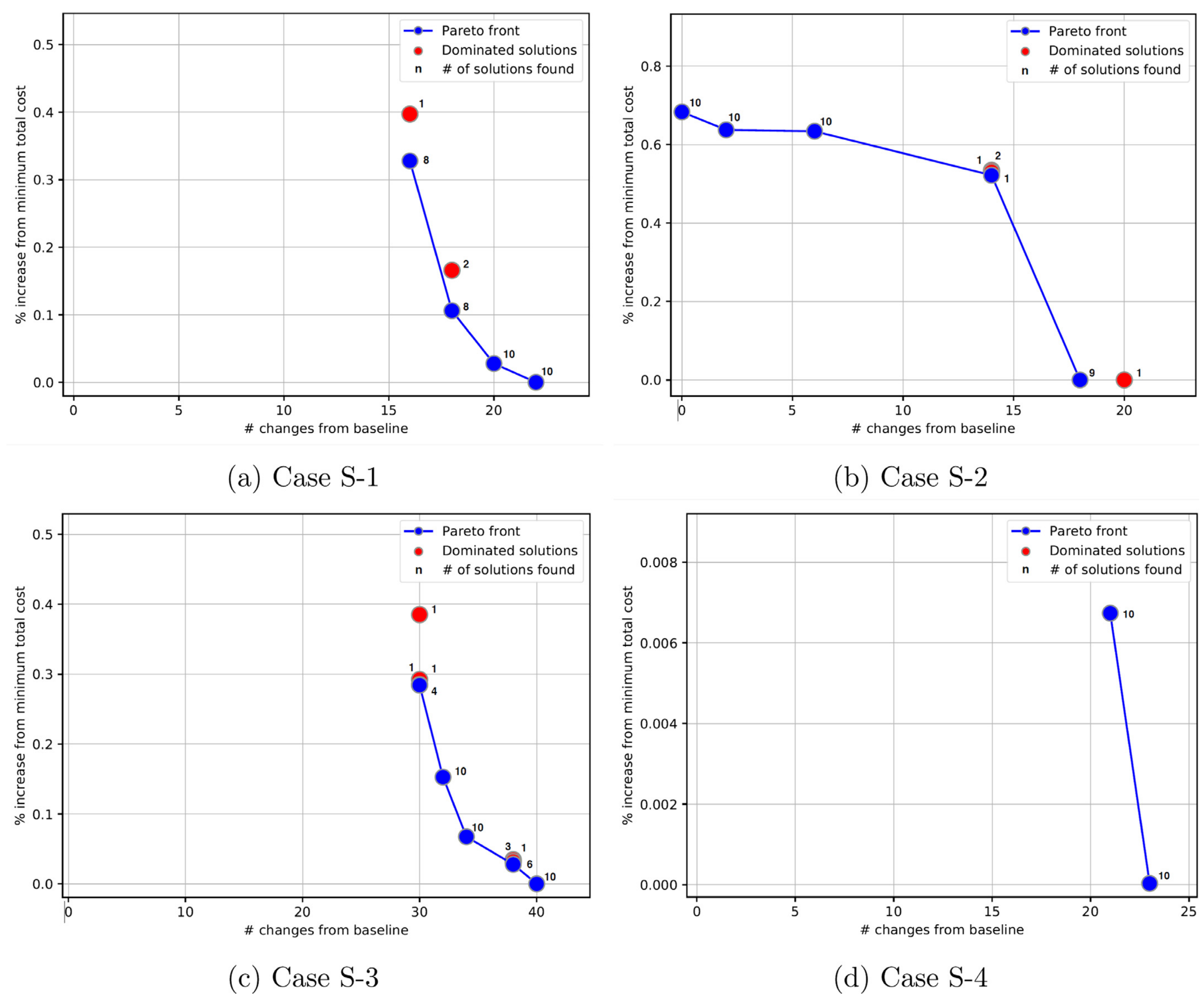


Fig. 3. The optimal Pareto front found by the ϵ-constraint method compared with the results from ten runs of the HGSADC algorithm for the four small test cases.

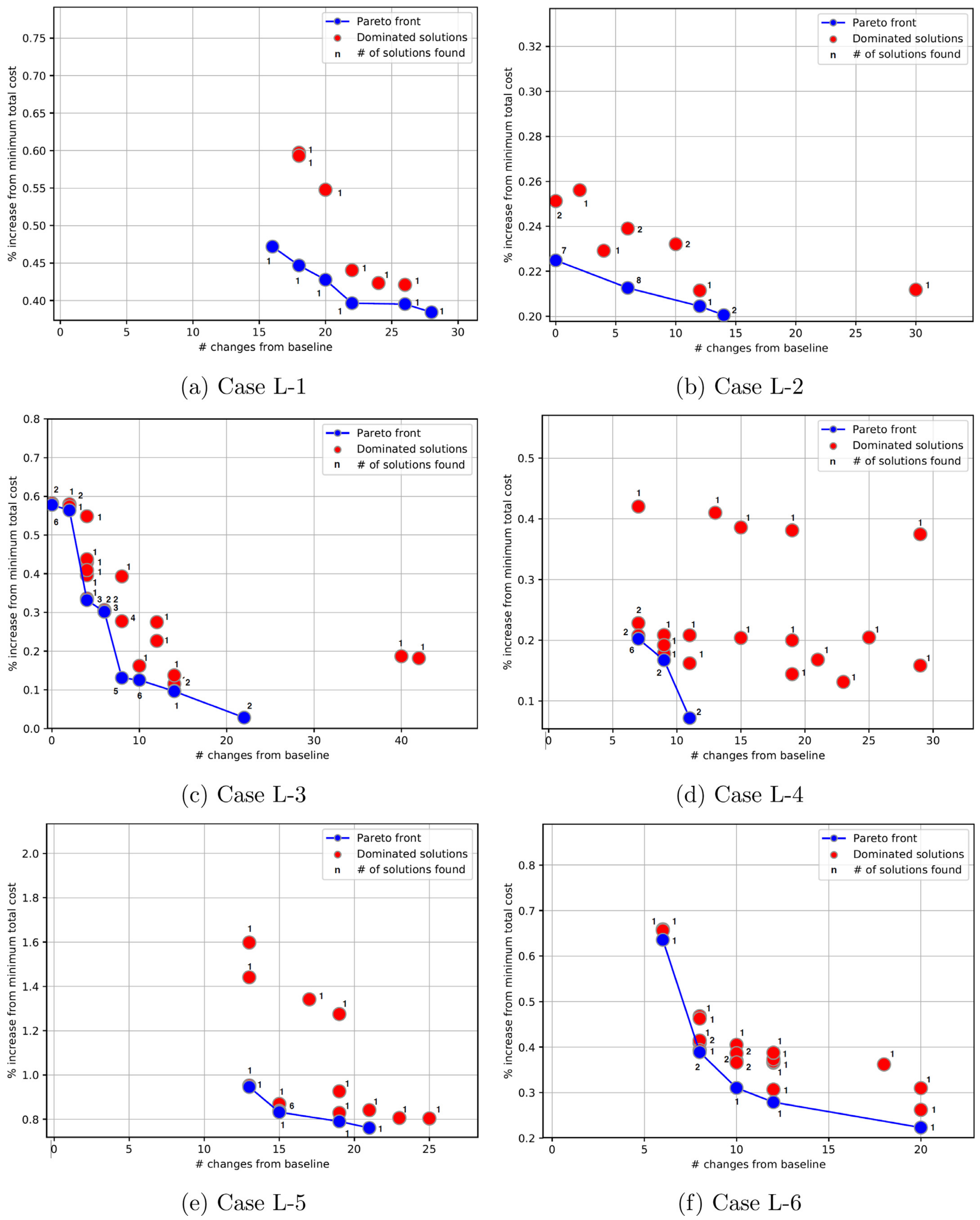
show the solutions that were part of the approximated Pareto front found by the HGASDC algorithm in at least one of the runs, but that were later dominated by the combined Pareto front. The numbers above each point show the number of times each solution was found among the ten runs. For all four cases, the combined fronts obtained among the ten runs of the HGASDC algorithm coincide with the optimal Pareto fronts obtained by the ϵ-constraint method, which show that the HGSADC algorithm was able to solve these cases to optimality.

We note that most of the HGASDC runs are able to find almost all solutions along the Pareto front (i.e., most of the numbers above the blue points are close to 10). Two exceptions here are Cases S-2 and S-3, where one of the Pareto-optimal solutions were found only one and four out of the ten runs, respectively. It can also be noted that even the dominated solutions that were found in some of the runs (the red points in Fig. 3) are good solutions that are very close to the optimal Pareto front.

In Case S-1, for example, the optimal Pareto front consists of four Pareto-optimal solutions. The rightmost Pareto-optimal point, where the total cost increase is 0 and the number of changes from baseline equals 22, represents the solution with the lowest possible total costs, i.e., the optimal solution with only cost minimization as objective. On the contrary, the leftmost Pareto-optimal point on the optimal front represents the solution with the lowest possible number of changes: 16 changes from the baseline solution. In other words, after taking persistence into account, the number of changes can be reduced from 22 to 16 by increasing the total costs by less than 0.35%. The decision as to whether it is worthwhile then rests with the planners.

It should be noted that the costs used in Fig. 3 (as well in Fig. 4) include the fixed time-charter costs for the PSVs (including crew), as these are the costs that are of most interest to the planners. Since the time-charter cost dominates the variable sailing (fuel) cost, the cost differences are rather small. By only looking at the variable sailing cost in Case S-1 for example, the cost difference between the original and the updated cost-optimal plan is 2.3% (instead of less than 0.35% including the time-charter costs).

Table 2 shows the run times for the ϵ-constraint method and the HGSADC algorithm to obtain the Pareto fronts in Fig. 3. For the HGSADC algorithm, we record the average run time over ten runs, and the coefficient of variation (CV) of the run time. The CV is calculated as the standard deviation divided by the mean.



(a) Case L-1 (b) Case L-2

(c) Case L-3 (d) Case L-4

(e) Case L-5 (f) Case L-6

Fig. 4. The combined Pareto fronts found by ten runs of the HGSADC algorithm for each of the six large test cases.

Table 2
Run times for the ϵ-constraint method and the HGSADC algorithm to obtain Pareto fronts for the small cases. Results for the HGSADC algorithm are based on ten runs.

	ϵ-constraint	HGSADC	
Test cases	Time (s)	Avg. time (s)	CV of time
Case S-1	142	110	34%
Case S-2	138	64	25%
Case S-3	183	106	18%
Case S-4	49	53	12%
Average	128	83	25%

Table 3
Run times for the HGSADC algorithm to obtain the Pareto fronts for the large cases. Results are based on ten runs.

	HGSADC	
Test cases	Avg. time (s)	CV of time
Case L-1	364	124%
Case L-2	192	30%
Case L-3	347	40%
Case L-4	243	26%
Case L-5	324	23%
Case L-6	600	46%
Average	345	48%

From Table 2 we observe that the HGSADC algorithm requires slightly less run time in general for solving the four small cases, needing on average 83 s in contrast to 128 s required by the ϵ-constraint method. However, note that we only compare the two methods on small cases with a maximum of 12 installations because this is the largest size that FICO Xpress is able to solve within a reasonable time (10,000 s). It has been shown in Borthen et al. (2018) that when attempting to solve single-objective SVPP problems with 13 or 14 installations, FICO Xpress only manages to find solutions with large optimality gaps (around 20%) after 10,000 s and no feasible solutions at all for even larger problems. When implementing the ϵ-constraint method, since the single-objective SVPP problem is not only solved once but must be solved iteratively, FICO Xpress is therefore not able to handle cases with more than 12 installations.

The HGSADC algorithm, on the other hand, has been shown to find high-quality Pareto fronts in reasonable time for all four small cases. Its performance is also stable: it finds approximately the same front in each run (Fig. 3); and a coefficient of variation of 25% indicates that its time consumption is also quite stable (Table 2).

5.3. Results on large real-size cases

When facing real-size problems with more than 20 installations, e.g., the large Cases L-1, L-2, L-3, L-4, L5, and L-5 in Table 1, the exact ϵ-constraint method becomes computationally impracticable. Therefore, only the HGSADC algorithm is run on these four cases.

Fig. 4 shows the combined Pareto fronts obtained among ten runs of the HGSADC algorithm for each of the large cases. The horizontal axis is the number of changes from the baseline solution, and the vertical axis shows the % increase in total costs from the lowest known total costs. The lowest known total costs are obtained by ten runs of the single-objective HGSADC algorithm (Borthen et al., 2018) for each case.

From the results of Cases L-2 and L-3 in Fig. 4, we see that the HGSADC algorithm manages to find solutions with zero changes from the baseline solutions, and increase in total costs of only around 0.2% and 0.6%, respectively. These are probably the most ideal solutions for the management of the installations, since they can expect the same visiting patterns from the depot as before. However, the Pareto fronts found by the HGSADC also provide other alternatives for the offshore planners if some changes from the baseline solution are acceptable.

In Fig. 4, we also observe that more dominated solutions are generated by the HGSADC algorithm for the large cases compared to the small ones shown in Fig. 3. However, most of these dominated solutions remain very close to the Pareto front. One exception here is Case L-4, where Fig. 4 shows several dominated solutions that are relatively far away from the Pareto front. Nevertheless, since every run is completed in a few minutes, this small performance instability can be circumvented by performing multiple runs and combining the fronts. Table 3 shows the average and CV of the run times for the HGSADC algorithm to obtain the Pareto fronts in Fig. 4. On average, HGSADC algorithm takes less than six minutes, with a coefficient of variation of 48%, to obtain a Pareto front for a problem of realistic size. This enables multiple runs of the algorithm in reality to achieve better Pareto fronts and a better decision support for offshore planners, since the frequency at which the problem needs to be solved is typically once very few weeks.

6. Conclusion

In this paper we have proposed a bi-objective optimization framework for the offshore supply vessel planning problem (SVPP) that simultaneously takes into account two objectives: minimizing cost and *minimizing changes from the baseline solution* (or maximizing persistence), so that the new solution deviates from the baseline solution as little as possible. We presented a bi-objective SVPP model and a genetic search approach for providing high-quality solutions, i.e., new weekly plans for the PSVs that are cost-efficient and close to the baseline solution. We tested the solution approach on cases of both small and large sizes based on real problems faced by Equinor. Through comparison with an exact ϵ-constraint method on four small cases, we showed that the proposed Hybrid Genetic Search Algorithm with Advanced Diversity Control (HGSADC) algorithm is able to provide high-quality Pareto fronts in reasonable time. For large real-size cases where the ϵ-constraint method is not applicable, the proposed HGSADC algorithm was shown to have the decision support capability in real life operations.

One interesting direction of future research is to modify the HGSADC algorithm to handle other objectives. For example, instead of, or in addition to, persistence, one can consider robustness of the weekly plan against unforeseen events as additional objectives.

Acknowledgments

The authors acknowledge financial support from project *Maritim Offshore Logistikk Optimering* (MOLO) partly funded by the Research Council of Norway. The last author is partially funded by Conselho Nacional de Desenvolvimento Científico e Tecnológico, (9 308528/2018-2), Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, (E-26/203.310/2016), Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro. The authors would also like to thank Tor Toftøy and Ellen Karoline Norlund at Equinor. The authors are grateful to the two reviewers, whose comments and suggestions helped us improve the paper.

References

Borthen, T., Loennechen, H., Wang, X., Fagerholt, K., Vidal, T., 2018. A genetic search-based heuristic for a fleet size and periodic routing problem with application to offshore supply planning. EURO J. Transp. Logist. 7 (2), 121–150. doi:10.1007/s13676-017-0111-x.

Brown, G.G., Cormican, K.J., Lawphongpanich, S., Widdis, D.B., 1997. Optimizing submarine berthing with a persistence incentive. Naval Res. Logist. 44 (4), 301–318. doi:10.1002/(SICI)1520-6750(199706)44:4<301::AID-NAV2>3.0.CO;2-A.

Brown, G.G., Dell, R.F., Farmer, R.A., 1996. Scheduling coast guard district cutters. Interfaces 26 (2), 59–72. doi:10.1287/inte.26.2.59.

Brown, G.G., Dell, R.F., Wood, R.K., 1997. Optimization and persistence. Interfaces 27 (5), 15–37. doi:10.1287/inte.27.5.15.

Coelho, L.C., Cordeau, J.-F., Laporte, G., 2012. Consistency in multi-vehicle inventory-routing. Transp. Res. Part C 24, 270–287.

Cordeau, J.-F., Laporte, G., Mercier, A., 2001. A unified tabu search heuristic for vehicle routing problems with time windows. J. Oper.l Res. Soc. 52 (8), 928–936. doi:10.1057/palgrave.jors.2601163.

Deb, K., 2014. Multi-objective optimization. In: Burke, E.K., Kendall, G. (Eds.), Search Methodologies. Springer US, Boston, MA, pp. 403–449. doi:10.1007/978-1-4614-6940-7_15.

Fagerholt, K., Korsvik, J.E., Løkketangen, A., 2009. Ship routing and scheduling with persistence and distance objectives. In: Bertazzi, L., Speranza, M.G., van Nunen, J. (Eds.), Innovations in Distribution Logistics. In: Lecture Notes in Economics and Mathematical Systems, vol. 619. Springer, Berlin, Heidelberg, pp. 89–107. doi:10.1007/978-3-540-92944-4_6.

Fagerholt, K., Lindstad, H., 2000. Optimal policies for maintaining a supply service in the Norwegian Sea. Omega 28 (3), 269–275. doi:10.1016/S0305-0483(99)00054-7.

Francis, P.M., Smilowitz, K.R., Tzur, M., 2008. The period vehicle routing problem and its extensions. In: Golden, B.L., Raghavan, S., Wasil, E.A. (Eds.), The Vehicle Routing Problem: Latest Advances and New Challenges. Springer US, Boston, MA, pp. 73–102. doi:10.1007/978-0-387-77778-8_4.

Halvorsen-Weare, E.E., Fagerholt, K., 2011. Robust supply vessel planning. In: Pahl, J., Reiners, T., Voß, S. (Eds.), Network Optimization. INOC 2011. In: Lecture Notes in Computer Science, vol. 6701. Springer, pp. 559–573. doi:10.1007/978-3-642-21527-8_62.

Halvorsen-Weare, E.E., Fagerholt, K., Nonås, L.M., Asbjørnslett, B.E., 2012. Optimal fleet composition and periodic routing of offshore supply vessels. Eur. J. Oper. Res. 223 (2), 508–517. doi:10.1016/j.ejor.2012.06.017.

Kisialiou, Y., Gribkovskaia, I., Laporte, G., 2018. Robust supply vessel routing and scheduling. Transp. Res. Part C 90, 366–378.

Kisialiou, Y., Gribkovskaia, I., Laporte, G., 2018. The periodic supply vessel planning problem with flexible departure times and coupled vessels. Comput. Oper. Res. 94, 52–64. doi:10.1016/j.cor.2018.02.008.

Konak, A., Coit, D.W., Smith, A.E., 2006. Multi-objective optimization using genetic algorithms: a tutorial. Reliabil. Eng. Syst. Saf. 91 (9), 992–1007. doi:10.1016/j.ress.2005.11.018.

Kovacs, A.A., Parragh, S.N., Hartl, R.F., 2015. The multi-objective generalized consistent vehicle routing problem. Eur. J. Oper. Res. 247, 441–458.

Mitchell, M., 1998. An Introduction to Genetic Algorithms. MIT Press.

Norlund, E.K., Gribkovskaia, I., 2013. Reducing emissions through speed optimization in supply vessel operations. Transp. Res. Part D 23, 105–113. doi:10.1016/j.trd.2013.04.007.

Norlund, E.K., Gribkovskaia, I., 2017. Environmental performance of speed optimization strategies in offshore supply vessel planning under weather uncertainty. Transp. Res. Part D 57, 10–22. doi:10.1016/j.trd.2017.08.002.

Norlund, E.K., Gribkovskaia, I., Laporte, G., 2015. Supply vessel planning under cost, environment and robustness considerations. Omega 57, 271–281. doi:10.1016/j.omega.2015.05.006.

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 50 (4), 195–204. doi:10.3138/infor.50.4.195.

Vidal, T., Crainic, T.G., Gendreau, M., Lahrichi, N., Rei, W., 2012. A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. Oper. Res. 60 (3), 611–624. doi:10.1287/opre.1120.1048.

Vidal, T., Crainic, T.G., Gendreau, M., Prins, C., 2014. A unified solution framework for multi-attribute vehicle routing problems. Eur. J. Oper. Res. 234 (3), 658–673. doi:10.1016/j.ejor.2013.09.045.

Vidal, T., Crainic, T.G., Gendreau, M., Prins, C., 2015. Time-window relaxations in vehicle routing heuristics. J. Heuristics 21 (3), 329–358. doi:10.1007/s10732-014-9273-y.