Transportation Research Part C 19 (2011) 853–865

Contents lists available at ScienceDirect

Transportation Research Part C

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

# Tramp ship routing and scheduling with speed optimization

Inge Norstad [a,b], Kjetil Fagerholt [a,*], Gilbert Laporte [c]

[a] Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology, Alfred Getz veg 3, NO-7491 Trondheim, Norway
[b] Norwegian Marine Technology Research Institute (MARINTEK), POB 4125 Valentinlyst, NO-7450 Trondheim, Norway
[c] Canada Research Chair in Distribution Management, HEC Montréal, 3000 chemin de la Côte-Sainte-Catherine, Canada H3T 2A7

ARTICLE INFO

*Article history:*
Received 11 January 2010
Received in revised form 30 April 2010
Accepted 1 May 2010

*Keywords:*
Maritime transportation
Speed optimization
Routing
Scheduling

ABSTRACT

Tramp shipping companies are committed to transport a set of contracted cargoes and try to derive additional revenue from carrying optional spot cargoes. Traditionally, models for ship routing and scheduling problems are based on fixed speed and a given fuel consumption rate for each ship. However, in real life a ship's speed is variable within an interval, and fuel consumption per time unit can be approximated by a cubic function of speed. Here we present the tramp ship routing and scheduling problem with speed optimization, where speed on each sailing leg is introduced as a decision variable. We present a multi-start local search heuristic to solve this problem. To evaluate each move in the local search we have to determine the optimal speed for each sailing leg of a given ship route. To do this we propose two different algorithms. Extensive computational results show that the solution method solves problems of realistic size and that taking speed into consideration in tramp ship routing and scheduling significantly improves the solutions.



## 1. Introduction

Seaborne transportation is one of the main freight transportation modes, and the only cost effective option for transportation of large volumes between the continents. Approximately eight billion tons of goods are carried by sea each year (UNCTAD, 2008). The maritime transportation industry has over the last decade experienced an increased awareness of the effect of fuel usage on operational costs and on environmental emissions. The impact of optimization in transportation planning in this industry can be significant given the very large volume of goods transported by sea each year.

We study a tramp ship routing and scheduling problem. A shipping company operating in the tramp shipping market will have some long-term contracts of affreightment (COAs) which specify a number of cargoes that the company is committed to transport. In addition to the contracted (mandatory) cargoes, there exists a spot market where cargo owners announce their transportation needs. These are optional cargoes that the shipping company can choose to carry if it finds it profitable and has sufficient fleet capacity to do so.

Each cargo, both mandatory and optional, consists of a given quantity to be transported from a given loading port to a given discharge port. For each cargo there is a time window that defines when the cargo can be loaded, and there is often a time window for the discharge port as well. Each spot cargo has a specified freight income rate that determines the revenue the shipping company will receive if the cargo is transported. The income derived from carrying the mandatory contract cargoes can be considered as fixed in this setting as there is no option not to carry them.

A typical tramp shipping company operates a heterogeneous fleet of ships having different load capacities, speeds, cargo handling equipment, operating costs and physical dimensions (length, beam width, draft, etc.). Due to the different

* Corresponding author.
*E-mail addresses:* inge.norstad@iot.ntnu.no (I. Norstad), kjetil.fagerholt@iot.ntnu.no (K. Fagerholt), gilbert@crt.umontreal.ca (G. Laporte).

0968-090X/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.trc.2010.05.001

properties of the ships, there may be compatibility constraints between ships and ports, and between ships and cargoes. For example, a small ship may not be able to carry a heavy cargo, and a large ship with deep draft may not enter a shallow port. There are costs associated with visiting a port, dependent on the ship, and there are also fuel costs associated with sailing.

Every ship in the fleet has a service speed that is traditionally used when the shipping company plans its routes and schedules for the fleet. However, in reality, the ship can sail at other speeds as well. Normally, a ship has a minimum and a maximum cruising speed which define the range of speeds at which it can actually travel. Fuel consumption, and hence the cost of sailing a given distance, is strongly dependent on speed. As shown by Ronen (1982), a cubic function provides a good estimation, within the practical speed range, of the relationship between fuel consumption per time unit and speed for cargo ships. A quadratic function can be shown to provide a good estimation of the relationship between fuel consumption per distance unit and speed.

Fig. 1 depicts the relationship between speed and fuel consumption for a particular liquefied natural gas carrier with load capacity of 150,000 $m^3$. The fuel consumption is in tonnes (t) per traveled nautical mile (M) and the speed $v$ is in knots (M/h). The feasible speed range is between $\underline{v} = 14.1$ and $\overline{v} = 22$ knots. For this interval the quadratic function $c(v) = 0.0036v^2 - 0.1015v + 0.8848$ has shown to provide a good estimate for the relationship between fuel consumption $c(v)$ and sailing speed $v$.

The objective in the tramp ship routing and scheduling problem is to maximize profit, i.e. total freight income minus operating costs. The decisions to be made in the planning are:

1. selecting which spot cargoes to carry,
2. allocating cargoes to ships,
3. determining optimal ship routes and
4. determining optimal ship schedules based on the optimal speed of each ship for each sailing leg.

These decisions are not independent of each other, so in order to find the optimal solution, they must be considered simultaneously. What is new in this problem description compared to traditional ship routing and scheduling problem formulations, is that we consider speed as a decision variable in the routing problem. This means that the sailing time and costs between a given pair of ports are not parameters with given values, but variables that depend on the chosen speed.

Note that several authors have also combined the determination of ship routes with speed decisions, albeit in different contexts from ours. Benford (1981) propose a simple procedure for determining the best mix of ships for a specific transport service. Perakis (1985) found that the model suggested by Benford (1981) imposed an artificial constraint and was then corrected. Unlike our problem, their formulation has only one speed variable per ship, resulting in each ship using the same speed during the whole planning period. Perakis and Papadakis (1987a), Perakis and Papadakis (1987b) consider a fleet deployment problem, where a given amount of cargo is to be transported from one loading port to one destination port within a year. The solution defines the laden and ballast speed for each ship and which ships that should be laid up. Again, each ship ends up using the same laden and ballast speed during the while planning period. Brown et al. (1987) and Bausch et al. (1998) both study liquid bulk shipping problems where the speed for each sailing leg is a decision variable. They generate all feasible ship schedules a priori and select the best combination of schedules by set partitioning. However, there are given dates for loading and discharging for each cargo, which means that the sailing speed for each leg is implicitly determined by the distance and time between the port calls on a given candidate route. Papadakis and Perakis (1989) study the problem of minimum-cost operation of a fleet of ships that has to carry a specific amount of cargo from a set of loadings ports to a set of unloading ports in a given time period. Unlike our problem there are no time windows within which a given cargo must be loaded or unloaded, hence the optimal laden and ballast speed will be the same for the entire schedule for each ship. Fagerholt (2001) considers a multi-ship pickup and delivery problem similar to ours, but with focus on soft time windows. All

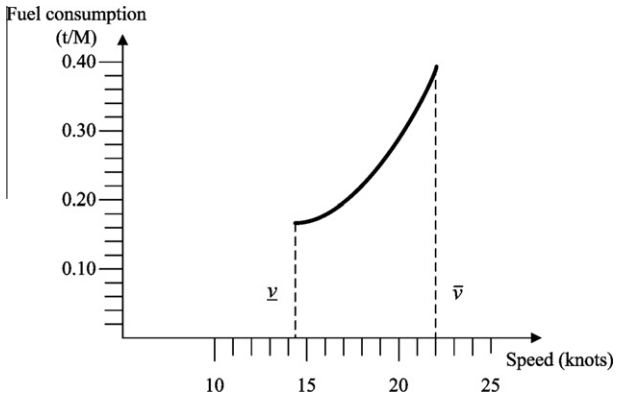


**Fig. 1.** Fuel consumption characteristics for an LNG carrier.

(promising) feasible routes are generated, and the optimal schedule for each route is then found by determining the optimal speed with respect to minimizing soft time window penalty and operating costs. Eventually, a set partitioning model selects the best combination of schedules. A drawback with this approach is that it cannot generate optimal solutions to large unconstrained problems where the number of feasible routes and schedules is too high to allow full enumeration. Ronen (forthcoming) studies the effect of oil price on the trade-off between reducing sailing speed and increasing the fleet size for container ships.

The purpose of this paper is threefold. First, we describe a way to incorporate speed decisions in a traditional tramp ship routing and scheduling problem. Second, we propose a solution method for this problem. Finally, we show that incorporating speed in ship routing and scheduling can yield significant improvements in profit for the shipping company.

The remainder of this paper is organized as follows. A mathematical formulation of the tramp ship routing and scheduling problem with speed optimization is given in Section 2. This section also provides a model for optimizing speed along a fixed single ship route, which arises as an important subproblem. In Section 3 we suggest solution methods, both for the speed optimization problem and the entire tramp ship routing and scheduling problem with speed optimization. Section 4 provides computational results, both for the subproblem and the global problem. Finally, concluding remarks are given in Section 5.

## 2. Mathematical model

We start by giving a mathematical formulation of the *Tramp Ship Routing and Scheduling Problem with Speed Optimization* (TSRSPSO) in Section 2.1, before we present a formulation of one of its important subproblems, namely the speed optimization problem (SOP) along a fixed single ship route.

### 2.1. The tramp ship routing and scheduling problem with speed optimization

A full mathematical arc flow formulation for the ship routing and scheduling problem with fixed speed is presented in Christiansen et al. (2007). It is formulated as a pickup and delivery problem with time windows and capacity constraints. There, as well as in most formulations of maritime transportation problems, the time and cost of sailing between a pair of ports for a given ship are fixed. Our extensions of that formulation lie in the introduction of new variables for the sailing speed for each ship and each sailing leg, as well as an adjusted cost function and constraints to incorporate speed as decision variables.

Let $\mathcal{N}$ be the set of cargoes indexed by $i$. Associated with cargo $i$ is a loading port node $i$ and a discharging port node $i+n$, where $n$ is the number of cargoes in $\mathcal{N}$. Let $\mathcal{N}_P = \{1,\dots,n\}$ be the set of loading (pickup) nodes and $\mathcal{N}_D = \{n+1,\dots,2n\}$, the set of delivery nodes. Note that several nodes can refer to the same physical port. The problem is defined on a graph $\mathcal{G}(\mathcal{N}_P \cup \mathcal{N}_D, \mathcal{A})$, where $\mathcal{A} = \{(i,j) : i,j \in \mathcal{N}_P \cup \mathcal{N}_D, i \neq j\}$. The set of loading nodes $\mathcal{N}_P$ is partitioned into two subsets, $\mathcal{N}_C$ and $\mathcal{N}_O$: $\mathcal{N}_C$ is the set of loading nodes for the mandatory contract cargoes, and $\mathcal{N}_O$ is the set of loading nodes for the optional cargoes. Let $\mathcal{V}$ be the set of ships indexed by $k$. The graph $\mathcal{G}_k(\mathcal{N}_k, \mathcal{A}_k)$ is the subgraph of $\mathcal{G}$ for ship $k$. Included in $\mathcal{N}_k$ are all the nodes of $\mathcal{N}_P \cup \mathcal{N}_D$ which correspond to cargoes that ship $k$ can carry. $\mathcal{N}_{Pk}$ and $\mathcal{N}_{Dk}$ are the sets of pickup and delivery nodes, respectively, for ship $k$. Included in $\mathcal{N}_k$ are also $o(k)$, the initial position for $k$, which can be a port or a point at sea, and $d(k)$, an artificial destination node. Ship $k$ has a feasible sailing speed interval $[\underline{v}_k, \overline{v}_k]$ and a cargo carrying capacity $Q_k$. Each cargo $i$ has a weight $q_i$ and generates a revenue $r_i$. For each node there is a time window $[\underline{t}_i, \overline{t}_i]$ which defines when service at the node must start. Let $s_{ik}$ denote the service time at node $i$ when visited by ship $k$. Let $d_{ij}$ be the sailing distance from node $i$ to node $j$.

The binary variable $x_{ijk}$ is 1 if ship $k$ travels directly from node $i$ to node $j$, and 0 otherwise. The variable $w_{ik}$ is the weight on board ship $k$ when leaving node $i$, and the variable $t_{ik}$ is the time for start of service for ship $k$ at node $i$. The variable $v_{ijk}$ is the speed of travel from node $i$ to node $j$ with ship $k$. The time it takes for ship $k$ to sail along arc $(i,j)$, including the service time $s_{ik}$ at $i$, is $d_{ij}/v_{ijk} + s_{ik}$. The non-linear function $c_k(v)$, defined in the interval $[\underline{v}_k, \overline{v}_k]$ represents the sailing costs per distance unit for ship $k$ sailing at speed $v$. The cost of sailing an arc $(i,j)$ with ship $k$ at speed $v_{ijk}$ is then $d_{ij}c_k(v_{ijk})$.

The TSRSPSO can now be formulated as follows:

$$\text{maximize} \sum_{k\in\mathcal{V}} \sum_{(i,j)\in\mathcal{A}_k} r_i x_{ijk} - \sum_{k\in\mathcal{V}} \sum_{(i,j)\in\mathcal{A}_k} d_{ij} c_k(v_{ijk}) x_{ijk}, \tag{2.1}$$

subject to

$$\sum_{k\in\mathcal{V}} \sum_{j\in\mathcal{N}_k} x_{ijk} = 1, \quad i \in \mathcal{N}_C, \tag{2.2}$$

$$\sum_{k\in\mathcal{V}} \sum_{j\in\mathcal{N}_k} x_{ijk} \leqslant 1, \quad i \in \mathcal{N}_O, \tag{2.3}$$

$$\sum_{j\in\mathcal{N}_{Pk}\bigcup\{d(k)\}} x_{o(k)jk} = 1, \quad k \in \mathcal{V}, \tag{2.4}$$

$$\sum_{i\in\mathcal{N}_k} x_{ijk} - \sum_{i\in\mathcal{N}_k} x_{jik} = 0, \quad k \in \mathcal{V}, j \in \mathcal{N}_k \setminus \{o(k), d(k)\}, \tag{2.5}$$

$$\sum_{i \in \mathcal{N}_{Dk} \bigcup \{o(k)\}} x_{id(k)k} = 1, \quad k \in \mathcal{V}, \tag{2.6}$$

$$\sum_{j \in \mathcal{N}_k} x_{ijv} - \sum_{j \in \mathcal{N}_k} x_{j,i+n,k} = 0, \quad k \in \mathcal{V}, i \in \mathcal{N}_{Pk}, \tag{2.7}$$

$$t_{ik} + s_{ik} + d_{i,i+n}/v_{i,i+n,k} - t_{i+n,k} \leqslant 0, \quad k \in \mathcal{V}, i \in \mathcal{N}_{Pk}, \tag{2.8}$$

$$x_{ijk}(t_{ik} + s_{ik} + d_{ij}/v_{ijk} - t_{jk}) \leqslant 0, \quad k \in \mathcal{V}, (i,j) \in \mathcal{A}_k, \tag{2.9}$$

$$\underline{t}_i \leqslant t_{ik} \leqslant \overline{t}_i, \quad k \in \mathcal{V}, i \in \mathcal{N}_k, \tag{2.10}$$

$$x_{ijk}(w_{ik} + q_j - w_{jk}) = 0, \quad k \in \mathcal{V}, j \in \mathcal{N}_{Pk}, (i,j) \in \mathcal{A}_k, \tag{2.11}$$

$$x_{i,j+n,k}(w_{ik} - q_j - w_{j+n,k}) = 0, \quad k \in \mathcal{V}, j \in \mathcal{N}_{Pk}, (i,j+n) \in \mathcal{A}_k, \tag{2.12}$$

$$w_{o(k)k} = 0, \quad k \in \mathcal{V}, \tag{2.13}$$

$$\sum_{j \in \mathcal{N}_k} q_i x_{ijk} \leqslant w_{ik} \leqslant \sum_{j \in \mathcal{N}_k} Q_k x_{ijk}, \quad k \in \mathcal{V}, i \in \mathcal{N}_{Pk}, \tag{2.14}$$

$$0 \leqslant w_{i+n,k} \leqslant \sum_{j \in \mathcal{N}_k} (Q_k - q_i) x_{i+n,j,k}, \quad k \in \mathcal{V}, i \in \mathcal{N}_{Pk}, \tag{2.15}$$

$$\underline{v}_k \leqslant v_{ijk} \leqslant \overline{v}_k, \quad k \in \mathcal{V}, (i,j) \in \mathcal{A}_k, \tag{2.16}$$

$$x_{ijk} \in \{0,1\}, \quad k \in \mathcal{V}, (i,j) \in \mathcal{A}_k. \tag{2.17}$$

The objective function (2.1) maximizes the profit from operating the fleet. Constraints (2.2) and (2.3) ensure that the contracted cargoes are transported exactly once, and that the spot cargoes are transported at most once, respectively. Constraints (2.4)–(2.6) describe the network flow on a route for ship $k$. Coupling constraints (2.7) ensure that the loading and discharging nodes for a cargo $i$ must be visited by the same ship. Precedence constraints (2.8) mean that the discharging node for a cargo must be visited after its loading node. Constraints (2.9) ensure that the time of starting service at a node $j$ must be greater than or equal to the departure time from the previous node $i$, plus the sailing time between the nodes. Constraints (2.10) define the time window in which service must start. Constraints (2.11) and (2.12) handle the relationship between quantity on board and loading and discharging, respectively, while constraints (2.13) define the initial quantity on board ship $k$. Constraints (2.14) and (2.15) ensure that the capacity constraints for the ships are not violated when loading and discharging, respectively. Constraints (2.16) are lower and upper bounds for the speed variables. Binary restrictions on the flow variables are given by constraints (2.17).

### 2.2. The speed optimization problem for a fixed ship route

This section describes a subproblem of the TSRSPSO, which is to determine the optimal speed along a fixed single ship route. We refer to this subproblem as the *Speed Optimization Problem* (SOP). It arises when solving the TSRSPSO by column generation or by local search. For both these methods, it is necessary to determine the profit contribution for each candidate route. The freight income and port costs for such a route will be fixed, but the fuel costs will depend on speed. In order to evaluate the correct profit for a given candidate route, one must determine the optimal speed for each leg of the sequence that will minimize the fuel cost for the route. Other authors, e.g. Sexton and Bodin (1985), Dumas et al. (1990), Ioachim et al. (1998), have studied a more general problem of schedule optimization on a fixed route, but with different objectives and constraints.

The problem description is based on the formulation presented by Fagerholt et al. (2010). Given a route, i.e. a sequence of port calls with time windows that a given ship must sail, the objective is to determine the speed for each leg in the route so that the total fuel consumption is minimized.

Fig. 2 illustrates the SOP along a given single ship route with four ports calls, or nodes. The node indexed by $i = 0$ is the initial position of the ship. The artificial destination node defined in Section 2.1 is not necessary in the case of a given route, hence it is not included in the figure.

For simplicity, we use the notation of Section 2.1, but, since we only consider a single ship route, the ship index $k$ is removed. The port call nodes along the route are indexed by $i = 0, \ldots, m$. Let the parameter $d_{i,i+1}$ represent the sailing distance along the arc $(i, i+1)$, defined for $i = 0, \ldots, m-1$. Each node along the route has a given time window $[\underline{t}_i, \overline{t}_i]$ in which service must start. Let $v_{i,i+1}$ be the decision variable for speed along arc $(i, i+1)$, and let $\underline{v}$ and $\overline{v}$ be the ship's minimum and maximum cruising speeds. Let $t_i$ denote the variable for the start of service time at node $i$. The non-linear cost function $c(v_{i,i+1})$ describes the relationship between fuel consumption and sailing speed. The objective is to minimize cost, since we can assume that the revenue from transporting the cargoes is fixed for the given route. In practice there will also be a specified service time at each node. This is, however, independent of sailing speed decisions, and hence we do not include this feature in the following model.

The SOP can now be defined as follows:

$$\text{minimize} \sum_{i=0}^{m-1} d_{i,i+1}\, c(v_{i,i+1}), \tag{2.18}$$

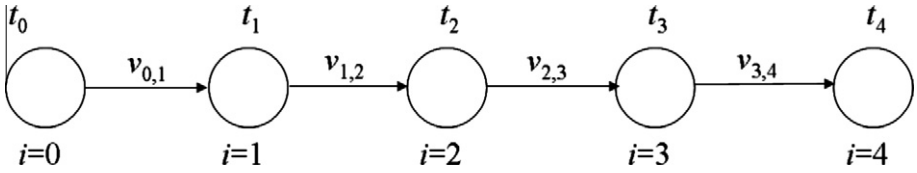


**Fig. 2.** Example of a single ship route with four port calls.

subject to

$$t_{i+1} - t_i - d_{i,i+1}/v_{i,i+1} \geqslant 0,\ i = 0,\ldots,m-1, \tag{2.19}$$

$$\underline{t}_i \leqslant t_i \leqslant \overline{t}_i,\ i = 1,\ldots,m, \tag{2.20}$$

$$\underline{v} \leqslant v_{i,i+1} \leqslant \overline{v},\ i = 0,\ldots,m-1. \tag{2.21}$$

The objective function (2.18) minimizes the fuel consumption for the given route. Constraints (2.19) ensure that the ship does not start service before it arrives at the node, but waiting is allowed. Constraints (2.20) are the time window constraints, while constraints (2.21) ensure that speed on a given arc is within the ship's feasible speed interval. For the start node, $i = 0$, there is no time window, only a start time $t_0$. We note that the objective function is convex and non-linear, and that constraints (2.19) are also non-linear.

## 3. Algorithms

In this section we present an algorithm for the TSRSPSO. The model can, in principle, be solved by commercial optimization software for non-linear programs, but in practice, real life instances of the problem will be too large to be solved within a reasonable time. Hence, we suggest a heuristic solution method. We will first introduce two alternative exact algorithms for the SOP, and we will then describe a heuristic for the TSRSPSO.

### 3.1. Solution methods for the SOP

The SOP can be solved by applying a non-linear programming (NLP) solver to the mathematical formulation of Section 2.2. However, even though the typical size of such a problem is quite small (four to 20 nodes), using an NLP solver will be too time consuming due to the large number of subproblems that need to be solved in the TSRSPSO algorithm. We will therefore present two alternative solution methods. The first discretizes the possible arrival times for each node along the route. The second method recursively adjusts the average speed for segments of the route until a feasible and optimal solution has been identified.

#### 3.1.1. Discretizing arrival times

This solution method is based on the work of Fagerholt et al. (2010) and Fagerholt (2001). We can discretize the arrival time within the time window of each node, and solve the problem as a *Shortest Path Problem* (SPP) on an acyclic graph. Consider the ship route illustrated in Fig. 2 and discretize the arrival times.

Each port call node in the route, except for $i = 0$, will be replicated into a number of discretized feasible arrival times at that node. Each of these new replicated nodes will represent a state $(i,l)$, where $i$ is the index of the node in the route, and $l$ refers to the discretized arrival time $t_{il}$ at the node. The variable $t_{01}$ is the start time for the ship. Each state $(i,l)$ will represent a node in a graph (see the example in Fig. 3). Every arc $((i,l),(i+1,l'))$ connects a state corresponding to an arrival time at a port call with a state corresponding to an arrival time at the next port call on the route. This results in a directed acyclic graph. Each arc will correspond to a given sailing speed, $v_{((il),(i+1,l'))} = d_{i,i+1}/(t_{i+1,l'} - t_{il})$. Arcs corresponding to speeds outside the feasible range for the ship and states with no incoming or outgoing arcs are not generated. To each arc is associated a sailing cost $c_{(il),(i+1,l')}$ that depends non-linearly on the corresponding speed of the arc. The sailing cost is the product of fuel consumption and fuel price. We note that $c_{(il),(i+1,l')} \geqslant c_{(il),(i+1,l'+1)}$, since the latter allows for slower sailing speed and hence less fuel consumption. This implies that for the last port call $m$, the latest arrival time will always be a part of the optimal solution, and thus all states for the last port call, except the latest arrival (node (4,3) in Fig. 3), could be removed from the graph. In practice, however, this could lead to a poor starting position for the following planning period. Although this is not done here, introducing a penalty cost for late arrival at the last port call is straightforward.

Every path through this graph, from node (0,1) to one of the states of the last port call node, is equivalent to a feasible ship schedule. Hence, finding a shortest path through the network is equivalent to finding an optimal arrival time for each port call. Then we also have determined the optimal speed for each leg of the route. Since the graph is acyclic and generated in a lexicographic order, the shortest path from the start node to any other node can effectively be computed while generating the network.

Discretizing speed instead of arrival times is another option which might even seem more intuitive. However, the graph would then become a tree where the number of nodes will grow exponentially with the number of discretizations and port calls on the route.

*3.1.2. A recursive smoothing algorithm*

We will now present a second algorithm that also can be used for determining vessel speed for each leg along the route. The idea behind this method is that since the fuel consumption function for a vessel is convex over the range of feasible speeds, it should be optimal to keep speed as low as possible and constant. For instance, if a route consists of two sailing legs, it will be better to keep the speed at 16 knots for both legs, than traveling at reduced speed on the first leg and increased speed on the second leg, even though the arrival time at the destination would be the same. Of course, the time window of the node between the two sailing legs may restrict this possibility. This property holds for all fuel consumptions that are function of speed as long as they are convex and non-decreasing over their defined interval. It is not necessary to know the coefficients of the function to determine the optimal speeds. This can be summarized in the following proposition.

**Proposition 1.** *Disregarding the time windows, the minimum cost speed $v^*$ for a given route with the sequence 0, 1, 2, ..., m, and a total sailing distance d, a given starting time $t_0$, and an ending time $t_m$, will be the same for all sailing legs and determined as $v^* = d/(t_m - t_0)$.*

**Proof.** The proof follows directly from the convex form of the fuel consumption (and hence cost) as a function of speed.

Our recursive smoothing algorithm (RSA) starts by considering the total time available from start to end of the route and the total distance to travel. Dividing the distance by the available time, we obtain the constant speed $v^*$ which is optimal in the absence of time windows. If none of the time windows is violated, the solution is optimal. If a time window is violated, the arrival time for that port call is adjusted to the nearest feasible value. Then the route is split at that node into two sub-routes and the procedure is repeated recursively until all time windows are respected. If the calculated speed in any iteration is greater than the ship's maximum speed, the problem is infeasible. The choice of the node on which to make the split can affect the objective value and the run time. We check all time windows and split the node $p$ with the largest time window violation $\delta$, no matter if the arrival is too early or too late.

Algorithm 1 describes the RSA algorithm. Let the variables $t_i$ be the arrival time at node $i$ and $v_{i,i+1}$ the speed from $i$ to $i + 1$. Let $s$ and $e$ be the start and end nodes for the (partial) route, respectively, and $d_{i,i+1}$ the distance from $i$ to $i + 1$. The time window for node $i$ is $[\underline{t}_i, \overline{t}_i]$. Before the algorithm is called, the start time and end time of the route must be set to $t_s = t_0$, which is the starting position for the ship, and $t_e = \overline{t}_e$, respectively.

**Algorithm 1** (*RECURSIVE_SMOOTHING_ALGORITHM* $(s, e)$).

$\delta \leftarrow 0$
$p \leftarrow 0$
$v^* \leftarrow \sum_{i=s}^{e-1} d_i/(t_e - t_s)$
**for** $i \leftarrow s$ to $e$ **do**
  $i \leftarrow i + 1$
  $v_i \leftarrow v^*$
  $t_i \leftarrow t_{i-1} + d_i/v_i$
  **if** $t_i - \overline{t}_i > |\delta|$ **then**
    $\delta \leftarrow t_i - \overline{t}_i$
    $p \leftarrow i$
  **end if**
  **if** $\underline{t}_i - t_i > |\delta|$ **then**
    $\delta \leftarrow t_i - \underline{t}_i$
    $p \leftarrow i$
  **end if**
**end for**
    **if** $\delta > 0$ **then**
  $t_p \leftarrow \overline{t}_p$
  RECURSIVE_SMOOTHING_ALGORITHM$(s,p)$
  RECURSIVE_SMOOTHING_ALGORITHM$(p,e)$
    **end if**
**if**$\delta < 0$ **then**
  $t_p \leftarrow \underline{t}_p$
  RECURSIVE_SMOOTHING_ALGORITHM$(s,p)$
  RECURSIVE_SMOOTHING_ALGORITHM$(p,e)$
**end if**

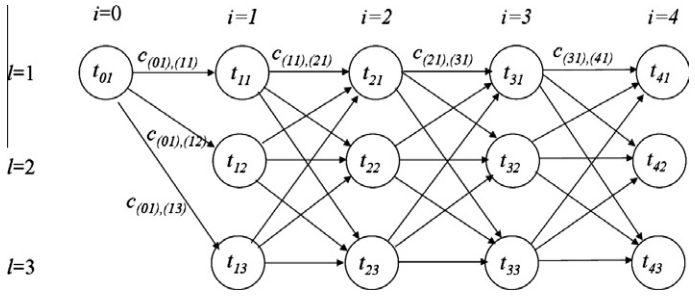


**Fig. 3.** Graph obtained by discretizing arrival times.

Before calculating the objective value for SOP, all optimal speeds $v_i$ that have a value lower than $\underline{v}$ must be set to $\underline{v}$. This means that the ship will sail at the lowest possible speed and wait for the start of the time window. However, one should note that the RSA algorithm is based on properties that only apply if the ship has the same fuel consumption function for all the sailing legs. If fuel consumption also depends on load, for example, then the property does not hold and the RSA cannot be used. If this is the case, the method using discretized arrival times should be used instead.

### 3.2. Algorithm for the TSRSPSO

To solve the tramp ship routing and scheduling problem we use a multi-start local search heuristic, following the ideas of Brønmo et al. (2007). In this heuristic a number of initial solutions are generated by a constructive heuristic. Each solution is generated partly randomly and partly by a deterministic insertion heuristic. Some cargoes are chosen randomly and each of them is assigned to a randomly chosen ship. The remaining cargoes are then assigned one by one to the ship that will contribute the most to the objective function. A number of the best initial solutions are then improved by a local search heuristic by means of intra-route operators, which attempt to improve the route for a single ship, and inter-route operators, which attempt to improve the solution by moving cargoes between ships.

Since this local search heuristic can be quite time consuming, not all initial solutions are improved, but only a number of the best ones. There are two local search procedures, a quick one and an extended one. The quick local search explores the neighborhood defined by three local search operators. In the last stage of the multi-start heuristic an extended local search is applied. The extended search uses two additional local search operators in addition to the three used in the quick search. Algorithm 2 gives an overview of the multi-start local search heuristic.

**Algorithm 2** (*MULTI-START*).

**Step 1:** Generate $\alpha$ initial solutions.
**Step 2:** Choose the $\beta$ best initial solutions and use the quick local search to improve them.
**Step 3:** Choose the $\gamma$ best solutions from Step 2 and improve them by using the extended local search.

Table 1 provides a brief description of the different local search operators. The quick local search uses the first three operators, while the extended local search uses all five. For more details on the multi-start local search heuristic, we refer to Brønmo et al. (2007). Whenever an operator is applied, the new solution obtained is evaluated with respect to feasibility and contribution to the objective function. In the local search heuristic, the maximum speed of the ship must be used in order to check for feasibility. Since the local search operators do not consider the speed decisions, each new ship route in a solution will have to be evaluated by solving the SOP. This subproblem minimizes voyage cost for a given ship route by determining the optimal speed for each leg in the route.

**Table 1**
Local search operators.

| Operator | Description |
|---|---|
| 1-resequence | A cargo $i$ is removed from the schedule of a ship $v$ and reinserted into the schedule of $v$ in the best possible position |
| Reassign | A cargo $i$ is removed from the schedule of a ship $v$ and inserted to the ship $u$ that gives the best feasible insertion |
| 2-interchange | Cargo $i$ is removed from ship $v$ and cargo $j$ is removed from ship $u$. Then $i$ is inserted into the schedule of ship $u$ and $j$ is assigned to ship $v$ |
| 2-resequence | Two cargoes $i$ and $j$ are removed from ship $v$. First cargo $i$ is inserted in the schedule for ship $v$ at the best position then cargo $j$ is reinserted in the best position |
| 3-interchange | Cargoes $i$,$j$ and $k$ are removed from ships $v$,$u$ and $w$, respectively. Cargoes $i$,$j$ and $k$ are then reinserted into the schedules of ship $u$,$w$ and $v$, respectively |

## 4. Computational study

We have performed extensive computational experiments in order to measure the effect of introducing speed as decision variables in the tramp ship routing and scheduling problem. In Section 4.1 we first study the SOP in isolation and compare the two proposed algorithms with the optimal solution provided by a non-linear mathematical programming solver. Then, in Section 4.2 we study the complete TSRSPSO.

### *4.1. Comparing SOP solution methods*

Fagerholt et al. (2010) solve a large number of generated problem instances. They compare the shortest path based method with the optimal solutions provided by a non-linear programming solver using an interior point method. In this paper we also compare the recursive smoothing algorithm (RSA) with the shortest path based method and with an exact algorithm.

#### *4.1.1. Generation of test problem instances for the SOP*

To test the performance of the different solution methods for the speed optimization problem, we have used an instance generator similar to the one used by Fagerholt et al. (2010). The same ship characteristics, i.e. speed range and fuel consumption function, are used for all the instances. In order to have a varied set of instances, three different parameters are set at different values. These parameters are:

1. Number of nodes in the route (4, 8, 12 or 16).
2. Amount of slack or waiting in the route (0%, 20% or 40%). This parameter determines the percentage of the port calls (rounded to the nearest integer) along the route that will have a waiting time for the start of the time window when using the ship's service speed. For the nodes chosen to have waiting, the time window will start between one and six (randomly chosen) days after what would have been the arrival time given sailing at service speed
3. The width of the time window (5 or 10 days).

The combination of four different route lengths, three alternative waiting time settings and two time window widths results in 24 different cases. Distances between nodes are randomly chosen from a realistic set of port to port distances. For nodes with waiting time, the start of the time window is given by the amount of waiting time. For the other nodes the start of the time windows are between 0.25 and 0.75 times the width of the time window earlier than the arrival time given service speed. Each case is labeled by using the values of the parameters, hence case 8n_10tw_40 refers to a case with eight nodes, 10 day wide time windows and waiting at 40% of the nodes. We have generated 100 instances of each case, resulting in 2400 different instances.

#### *4.1.2. Computational results for the SOP*

The 2400 instances were solved by applying the freely available non-linear programming solver IPOPT from COIN-OR, to the mathematical formulation of Section 2.2. An algorithm for solving the model with discretized arrival times was implemented in C#, as described by Fagerholt et al. (2010). For each instance an acyclic network is generated and a shortest path is determined. The RSA is also implemented in C#. Table 2 shows the optimality gap for the different algorithms applied on the 24 cases. The entries in Table 2 are average values over 100 instances for each case. SPP $x$ refers to the shortest path based algorithm with a discretization level of $x$, whereas RSA refers to the recursive smoothing algorithm.

To compare the response time of the solution methods, we have generated a large number of test instances of four different route lengths and solved them with the different methods. Table 3 shows the average CPU times in milliseconds for the different algorithms and the different route lengths. Even though the non-linear programming optimization software solves the instances in less than a second, it is too slow to be used as a move evaluator in the local search based heuristic for the TSRPSO. The SPP method, on the other hand, is much faster and more practical. We see that the response time for the SPP grows quickly when the discretization level increases, but it is linearly dependent on route length. When we take the results from Table 2 into account, we see that SPP5 or SPP10 would probably be sufficient for the evaluation of each route when solving the TSRPSO. At this level the CPU time is acceptable. The RSA is extremely fast and always provides optimal solutions. Therefore, for ship routing problems in which each ship has the same fuel consumption function for both laden and ballast voyages, this algorithm is the obvious choice.

We should recall that the fuel consumption function we are using is just an estimate, and that several factors are not known at the time of planning, such as weather, that will influence the fuel consumption. Bearing this in mind, one should not be too focused on the optimality gap, as there is much uncertainty. Also, during the local search heuristic, what is important is that the best routing decision should be taken. The final speed decisions can be taken a posteriori. For all practical purposes, a solution value less than 1% away from the optimal value should be sufficient. A good solution strategy for the TSRSPSO, if each ship has different fuel consumption functions depending on the ship load, could therefore be to use SPP5 or SPP10 for the evaluation of each move, and SPP100 or a non-linear programming software for the final solution. When fuel consumption function per ship only depends on speed, RSA is clearly the best algorithm for the SOP.

**Table 2**
Average optimality gap (in percent) for different SOP solution methods.

| Case | SPP5 | SPP10 | SPP20 | SPP50 | SPP100 | RSA |
|---|---|---|---|---|---|---|
| 4n_5tw_0 | 0.418 | 0.072 | 0.018 | 0.003 | 0.001 | 0.000 |
| 4n_5tw_20 | 0.180 | 0.038 | 0.008 | 0.001 | 0.000 | 0.000 |
| 4n_5tw_40 | 0.112 | 0.019 | 0.005 | 0.001 | 0.000 | 0.000 |
| 4n_10tw_0 | 1.082 | 0.188 | 0.048 | 0.007 | 0.002 | 0.000 |
| 4n_10tw_20 | 0.526 | 0.140 | 0.024 | 0.004 | 0.001 | 0.000 |
| 4n_10tw_40 | 0.261 | 0.070 | 0.014 | 0.002 | 0.000 | 0.000 |
| 8n_5tw_0 | 0.838 | 0.144 | 0.029 | 0.004 | 0.001 | 0.000 |
| 8n_5tw_20 | 0.640 | 0.104 | 0.023 | 0.003 | 0.001 | 0.000 |
| 8n_5tw_40 | 0.253 | 0.052 | 0.010 | 0.002 | 0.000 | 0.000 |
| 8n_10tw_0 | 2.413 | 0.494 | 0.095 | 0.014 | 0.004 | 0.000 |
| 8n_10tw_20 | 1.413 | 0.257 | 0.063 | 0.009 | 0.002 | 0.000 |
| 8n_10tw_40 | 0.715 | 0.148 | 0.030 | 0.005 | 0.001 | 0.000 |
| 12n_5tw_0 | 1.078 | 0.164 | 0.038 | 0.006 | 0.001 | 0.000 |
| 12n_5tw_20 | 0.563 | 0.101 | 0.023 | 0.003 | 0.001 | 0.000 |
| 12n_5tw_40 | 0.261 | 0.050 | 0.012 | 0.002 | 0.000 | 0.000 |
| 12n_10tw_0 | 2.910 | 0.528 | 0.117 | 0.017 | 0.004 | 0.000 |
| 12n_10tw_20 | 1.799 | 0.313 | 0.071 | 0.011 | 0.003 | 0.000 |
| 12n_10tw_40 | 0.965 | 0.195 | 0.045 | 0.007 | 0.002 | 0.000 |
| 16n_5tw_0 | 1.176 | 0.200 | 0.042 | 0.006 | 0.002 | 0.000 |
| 16n_5tw_20 | 0.597 | 0.104 | 0.023 | 0.003 | 0.001 | 0.000 |
| 16n_5tw_40 | 0.327 | 0.066 | 0.013 | 0.002 | 0.000 | 0.000 |
| 16n_10tw_0 | 3.545 | 0.628 | 0.133 | 0.020 | 0.005 | 0.000 |
| 16n_10tw_20 | 1.963 | 0.332 | 0.075 | 0.011 | 0.003 | 0.000 |
| 16n_10tw_40 | 1.081 | 0.198 | 0.046 | 0.006 | 0.002 | 0.000 |
| **Average** | **1.047** | **0.192** | **0.042** | **0.006** | **0.002** | **0.000** |

**Table 3**
Average CPU time (in milliseconds) for different SOP solution methods.

| Nodes | NLP | SPP5 | SPP10 | SPP20 | SPP50 | SPP100 | RSA |
|---|---|---|---|---|---|---|---|
| 4 | 332 | 0.030 | 0.094 | 0.29 | 1.72 | 6.30 | 0.0010 |
| 10 | 401 | 0.078 | 0.188 | 0.70 | 4.12 | 16.2 | 0.0050 |
| 20 | 430 | 0.125 | 0.375 | 1.39 | 8.23 | 32.2 | 0.0091 |
| 40 | 577 | 0.281 | 0.750 | 2.81 | 16.81 | 64.7 | 0.0190 |

### *4.2. Multi-start local search algorithm*

The multi-start local search heuristic was coded in C# and implemented as a solver on the commercial ship routing decision support system TurboRouter (Fagerholt, 2004; Fagerholt and Lindstad, 2007) from MARINTEK. Brønmo et al. (2007) describe two different parameter settings for the multi-start heuristic, one with short computational times, and one that requires more computational time but provides better solutions. In the computational study here we will only use the latter parameter setting. The computational tests were performed on a PC with Pentium M, 1.8 GHz processor and 512 Mb of RAM under Windows XP. For each of the test cases we will compare different strategies for solving the problems.

#### *4.2.1. Description of the test problem instances for TSRSPSO*

We have used two different sets of problem instances. First, we have nine real cases with data from six different international bulk shipping companies operating in the tramp shipping market. The first eight are the same instances that have been studied in Korsvik et al. (2010) and Brønmo et al. (2007). The ships in instances one to six may carry multiple cargoes simultaneously, while in instances seven to nine all cargoes are full ship loads. Table 4 describes the nine instances. In the original cases, each ship has a given service speed and a fuel consumption. In order to solve the instances as TSRSPSO, the service speed is replaced by a minimum and a maximum speed, and the fixed fuel consumption is replaced by a fuel consumption function depending on speed.

In addition to these instances we have developed an instance generator that can generate instances with different characteristics. The fleet is based on real ships and is the same for all the generated problem instances. The fleet has 13 ships with different load capacities and speed intervals. The cargoes are generated by randomly selecting a loading port and a discharging port from a set of real ports. The cargo sizes are set randomly between 20% and 100% of the capacity of the largest ship, hence carrying multiple cargoes simultaneously is possible. Time window widths are chosen randomly between 6 and 14 days, and the freight rates are randomly chosen within an interval representing realistic figures. We have designed four different cases and generated 10 instances of each, resulting in 40 different instances. Table 5 describes the four instance sets.

**Table 4**
Test instance descriptions.

| Instance | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| Planning horizon (days) | 23 | 75 | 75 | 40 | 35 | 150 | 20 | 35 | 90 |
| # of contract cargoes | 18 | 8 | 17 | 12 | 15 | 41 | 28 | 12 | 16 |
| # of spot cargoes | 0 | 1 | 0 | 2 | 2 | 9 | 2 | 3 | 2 |
| # of ships | 6 | 3 | 6 | 7 | 13 | 13 | 13 | 4 | 6 |

*4.2.2. Computational results*

To measure the effect of introducing speed decisions in the ship routing problem, we have compared different strategies for taking the speed decisions into account. Table 6 shows the results from applying different strategies on test instance number nine. In this problem there are 18 cargoes and a heterogeneous fleet of six ships, all with a service speed of 17 knots, and 18 cargoes. In the first column we have solved the problem with a traditional fixed service speed formulation. In the next column we have taken the solution from the first one and have applied the RSA algorithm a posteriori to each of the six shipping routes. We see that the income is the same, but the fuel consumption is significantly reduced when speed is optimized. We also see that only 16 of the 18 cargoes are transported. In the third column the problem is solved as a fixed speed problem, with all ships traveling at the maximal speed of 20 knots. Now, all the 18 cargoes are taken, but due to the constant high speed, the fuel costs are so high that the profit is less than under the previous strategies. In the fourth column we have optimized the speed a posteriori on the routing solution found in the third column. Finally, in the fifth column we have applied the RSA to solve the SOP as a move evaluator in the multi-start local search. The income is the same, since all the 18 cargoes are taken, but because the routing is different, the fuel consumption is 4.3% less and hence the profit is 2.9% higher.

In Table 7 we solve all nine real instances with each of the five different solution strategies. For each instance we report the gap in percent for both net profit and fuel costs compared with the fixed service speed solution. We see that for all the instances, applying speed optimization during the evaluation of each move in the local search heuristic provides the best solution. On average the improvement in profit is 7.0%. The effect of applying speed optimization varies from instance to instance and is highly dependent on several factors, for example on the relationship between the number of cargoes and fleet capacity, and on the relationship between fuel price and freight rates. Note that in instance two, solving the problem by applying a fixed maximum speed increases the profit. This is because speeding up allows the fleet to transport an additional spot cargo, and hence increases the income. This is also the reason why fuel consumption increases when applying the maximum speed with speed optimization and variable speed strategies to instances two and nine.

In Table 8 we solve the 40 generated instances with each of the five different solution strategies. The values in the table are average results over the 10 instances of each set. We see that on average, over all the 40 instances, there is an 14% improvement in net profit when applying speed optimization to evaluate each move in the local search, compared to the fixed service speed strategy.

Note that for instance set D we see an improvement in profit from service speed with speed optimization to maximum speed with speed optimization, while for sets A to C it is the opposite. In set D, the planning horizon is relatively short with respect to the number of cargoes. Therefore, on average over the 10 instances, only 66 of the 70 cargoes are taken when using service speed. By applying maximum speed, the average is 68 cargoes. This leads to an increased income, hence to an improvement from 5.7% to 8.7% in profit. In instance sets A to C, due to the relatively long planning horizon, it is possible

**Table 5**
Descriptions of sets of generated problem instances.

| Instance set | A | B | C | D |
|---|---|---|---|---|
| Planning horizon (days) | 30 | 60 | 100 | 120 |
| # of contract cargoes | 0 | 0 | 0 | 0 |
| # of spot cargoes | 15 | 25 | 50 | 70 |
| # of ships | 13 | 13 | 13 | 13 |

**Table 6**
Comparing different strategies for solving instance number nine.

| | Service speed (17 knots) | Service speed + speed opt. | Max speed (20 knots) | Max speed + speed opt. | Variable speed (14–20 knots) |
|---|---|---|---|---|---|
| Cargoes served | 16 | 16 | 18 | 18 | 18 |
| Income ($) | 16.102.500 | 16.102.500 | 19.072.500 | 19.072.500 | 19.072.500 |
| Port costs($) | 2.837.050 | 2.837.050 | 3.351.450 | 3.351.450 | 3.351.450 |
| Fuel costs ($) | 5.830.661 | 5.184.264 | 10.188.202 | 6.328.365 | 6.056.490 |
| Profit ($) | 7.434.789 | 8.081.186 | 5.532.848 | 9.392.685 | 9.664.560 |

**Table 7**
Profit and fuel cost gap (in percent) for different strategies compared to fixed service speed

| Instance | Service speed + speed opt. | | Max speed | | Max speed + speed opt. | | Variable speed | |
|---|---|---|---|---|---|---|---|---|
| | Profit | Fuel | Profit | Fuel | Profit | Fuel | Profit | Fuel |
| 1 | 0.4 | −2.4 | −1.6 | 31.7 | 1.8 | −15.3 | 2.2 | −21.3 |
| 2 | 0.5 | −3.1 | 1.9 | 53.7 | 6.4 | 27.6 | 7.7 | 20.1 |
| 3 | 2.0 | −14.0 | −7.0 | 45.6 | 2.0 | −14.0 | 2.0 | −14.0 |
| 4 | 4.0 | −24.6 | −7.9 | 46.2 | 4.0 | −24.4 | 6.6 | −38.3 |
| 5 | 1.7 | −9.7 | −6.1 | 35.4 | 2.0 | −11.5 | 2.1 | −12.4 |
| 6 | 6.8 | −43.9 | −6.3 | 40.1 | 7.2 | −45.2 | 10.9 | −70.0 |
| 7 | 0.8 | −10.4 | −1.9 | 24.9 | 1.1 | −14.2 | 1.3 | −18.0 |
| 8 | 0.4 | −7.9 | −1.7 | 31.8 | 0.4 | −7.9 | 0.5 | −8.8 |
| 9 | 8.7 | −11.1 | −25.6 | 74.7 | 26.3 | 8.5 | 30.0 | 3.9 |
| **Average** | **2.7** | **−14.1** | **−6.2** | **42.7** | **5.7** | **−10.7** | **7.0** | **−17.6** |

**Table 8**
Profit and fuel cost gap (in percent) for different strategies compared to fixed service speed.

| Instance set | Service speed + speed opt. | | Max speed | | Max speed + speed opt. | | Variable speed | |
|---|---|---|---|---|---|---|---|---|
| | profit | fuel | profit | fuel | profit | fuel | profit | fuel |
| A | 12.1 | −33.1 | −13.7 | 43.2 | 9.3 | −29.6 | 16.4 | −52.4 |
| B | 11.5 | −34.9 | −13.1 | 39.8 | 1.1 | −3.4 | 12.6 | −38.4 |
| C | 15.2 | −58.1 | −10.2 | 38.7 | 5.6 | −21.5 | 13.7 | −51.9 |
| D | 5.7 | −22.1 | −6.4 | 48.4 | 8.7 | −9.9 | 13.2 | −28.9 |
| **Average** | **11.1** | **−37.1** | **−10.9** | **42.5** | **6.2** | **−16.1** | **14.0** | **−42.9** |

to transport all the cargoes when using the service speed. When applying the maximum speed, the solutions tend to utilize the ships more unevenly, which leads to a poorer result, compared to the service speed, when the speed optimization is applied a posteriori. We will now explain why this happens.

In a poor tramp ship market, the number of spot cargoes is relatively low, and a shipping company will have excess fleet capacity. In this case, using fixed speed will often lead to a solution in which the ships are unevenly utilized. This is because, when the speed is fixed, fuel consumption is proportional to distance sailed. Using fixed speed when assigning ships to the cargoes will in practice result in the minimization of the total traveled distance. A posteriori speed optimization will reduce the fuel cost, but will not change the routing decision. However, if variable speed is taken into account when assigning ships to the cargoes, the solution often includes a more even distribution of the cargoes among the ships, leading to lower average speed for the fleet, and hence lower fuel consumption. In the example in Table 9 there are two ships and six cargoes. The first column shows the optimal cargo assignment when the problem is solved by using maximum speed with a posteriori speed optimization. The second column shows the solution when solving the SOP for evaluating each move in the local search heuristic. Note that the total traveled distance is shorter in the fixed speed solution, but since the average speed is much lower in the variable speed solution, the total fuel consumption will be less in the latter.

### 4.2.3. Computational time

Table 10 shows the computational times associated with the different strategies for solving the TSRSPSO. The values are average values of all the 10 generated instances of each case. The time spent on applying speed optimization a posteriori is negligible compared with the time spent on the multi-start local search based heuristic. Hence we do not report response times for the pure fixed speed strategies, but only the fixed speed with a posteriori speed optimization. The column labeled *Service speed* refers to the service speed with a posteriori speed optimization, and *Max speed* refers to the maximum speed

**Table 9**
Example of different routing decisions.

| | Max speed + speed opt. | | Variable speed | |
|---|---|---|---|---|
| | Ship A | Ship B | Ship A | Ship B |
| Cargo 1 | ✓ | | | ✓ |
| Cargo 2 | | ✓ | ✓ | |
| Cargo 3 | ✓ | | | ✓ |
| Cargo 4 | | ✓ | ✓ | |
| Cargo 5 | | ✓ | ✓ | |
| Cargo 6 | | ✓ | | ✓ |
| Average speed (knots) | 14.0 | 16.3 | 14.2 | 14.0 |
| Distance (M) | 32.295 | | 32.869 | |
| Fuel cost ($) | 628.168 | | 556.774 | |

**Table 10**
Average response times (in seconds) for different strategies used to solve the TSRSPSO.

| Instance set | # cargoes | Service speed + speed opt. | Max speed + speed opt. | SPP | RSA |
|---|---|---|---|---|---|
| A | 15 | 8 | 10 | 20 | 11 |
| B | 25 | 18 | 20 | 39 | 22 |
| C | 50 | 282 | 571 | 967 | 599 |
| D | 70 | 340 | 840 | 1121 | 841 |

with a posteriori speed optimization. The column SPP refers to applying the discretized arrival times method with a discretization level of 10 for each move in the local search and a discretization level of 100 on the final solution. The column labeled RSA refers to applying the RSA algorithm to evaluate each move in the local search. For all the instances and strategies the multi-start parameters are the same: 100 initial solutions, 10 quick local searches and five extended searches. When applying a local search heuristic, it is impossible to predict from the size of the instance how many iterations will be needed to find a local optimum. Therefore, the computation time varies significantly from instance to instance and from solution method to solution method. For several of the instances, applying RSA to each local search move resulted in shorter computational time before an optimum was found. From Table 10 we see that on average solving the TSRSPSO with maximum speed require more time than with service speed. This is because the neighborhood size defined by the local search operators increases as a higher speed allows a ship to transport more cargoes. We see that on average over the four cases applying SPP10 requires 63% more time than applying RSA. We also see that applying RSA hardly requires any additional time compared to the Max speed strategy.

## 5. Concluding remarks

We have introduced and solved a tramp ship routing and scheduling problem in which speed is a decision variable on the different sailing legs. Since fuel consumption per distance unit can be approximated by a quadratic function of speed, the problem becomes non-linear. We have presented two algorithms for the speed optimization problem along a single shipping route. Discretizing arrival times is quite fast and can be applied to any fuel consumption function. By adjusting the parameter of discretization level, one can trade off between solution quality and solution time. The recursive smoothing algorithm (RSA) is even faster, but is only applicable for cases where fuel consumption functions of the ships are not dependent on ship load, but only on speed. It will, however, work for any convex fuel consumption function. Taking variable speed into consideration significantly improves the profit, partly because increasing speed can make it possible to carry additional spot cargoes, and partly because reducing speed results in less fuel consumption per distance. Reducing fuel consumption will also result in reduced environmental emissions, see also the survey of the environmental effects from optimization in land based transportation by Sbihi and Eglese (2007).

## Acknowledgements

This research was carried out with financial support from the DESIMAL project, funded by the Research Council of Norway and from the Canadian Natural Engineering Research Council under Grant 39682-05. This support is gratefully acknowledged. Thanks are due to the referees for their valuable comments.

## References

Bausch, D., Brown, G., Ronen, D., 1998. Scheduling short-term marine transport of bulk products. Maritime Policy and Management 25, 335–348.
Benford, H., 1981. A simple approach to fleet deployment. Maritime Policy & Management 8, 223–228.
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, 884–899.
Brown, G., Graves, G., Ronen, D., 1987. Scheduling ocean transportation of crude oil. Management Science 33, 335–346.
Christiansen, M., Fagerholt, K., Nygreen, B., Ronen, D., 2007. Maritime transportation. In: Barnhart, C., Laporte, G. (Eds.), Transportation. In: Handbooks in Operations Research and Management Science, vol. 14. North-Holland, Amsterdam, pp. 189–284.
Dumas, Y., Soumis, F., Desrosiers, J., 1990. Optimising the schedule for a fixed vehicle path with convex inconvenience costs. Transportation Science 24, 145–152.
Fagerholt, K., 2001. Ship scheduling with soft time windows – an optimization based approach. European Journal of Operational Research 131, 559–571.
Fagerholt, K., 2004. A computer-based decision support system for vessel fleet scheduling – experience and future research. Decision Support Systems 37, 35–47.
Fagerholt, K., Laporte, G., Norstad, I., 2010. Reducing fuel emissions by optimizing speed on shipping routes. Journal of the Operational Research Society 61, 523–529.
Fagerholt, K., Lindstad, H., 2007. Turborouter: an interactive optimization-based decision support system for ship routing and scheduling. Maritime Economics & Logistics 9, 214–233.
Ioachim, I., GTlinas, S., Soumis, F., Desrosiers, J., 1998. A dynamic programming algorithm for the shortest path problem with time windows and linear node costs. Networks 31, 193–204.
Korsvik, J.E., Fagerholt, K., Laporte, G., 2010. A tabu search heuristic for ship routing and scheduling. Journal of the Operational Research Society 61, 594–603.
Papadakis, N.A., Perakis, A.N., 1989. A nonlinear approach to multiorigin, multidestination fleet deployment problem. Naval Research Logistics 36, 515–528.
Perakis, A.N., 1985. A second look at fleet deployment. Maritime Policy & Management 12, 209–214.

Perakis, A.N., Papadakis, N.A., 1987a. Fleet deployment models, part 1. Maritime Policy & Management 14, 127–144.
Perakis, A.N., Papadakis, N.A., 1987b. Fleet deployment models part 2. Maritime Policy & Management 14, 145–155.
Ronen, D., 1982. The effect of oil price on the optimal speed of ships. Journal of the Operational Research Society 33, 1035–1040.
Ronen, D., forthcoming. The effect of oil price on containership speed and fleet size. Journal of the Operational Research Society.
Sbihi, A., Eglese, R.W., 2007. Combinatorial optimization and green logistics. 4OR 5, 99–116.
Sexton, T.R., Bodin, L.D., 1985. Optimizing single vehicle many-to-many operations with desired delivery times: I. Scheduling. Transportation Science 19, 378–410.
UNCTAD, 2008. Review of maritime transport, 2008. United Nations, New York and Geneva.