Omega 109 (2022) 102580

Contents lists available at ScienceDirect

Omega

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

Optimizing fuel consumption on inland waterway networks: Local search heuristic for lock scheduling☆

Julian Arthur Pawel Golak[a,b,*], Christof Defryn[a], Alexander Grigoriev[a]

[a] Maastricht University School of Business and Economics, PO Box 616, Maastricht 6200 MD, the Netherlands
[b] Hamburg University of Technology, Institute for Algorithms and Complexity, Hamburg, Germany

ARTICLE INFO

Article history:
Received 9 March 2021
Accepted 30 November 2021
Available online 17 January 2022

2010 MSC:
00-01
99-00

Keywords:
AIS Data
Inland waterway operations
Local search heuristic
Mixed-Integer programming

ABSTRACT

Fuel consumption and CO_2 emission are among the main criteria to assess the environmental and economical impact of vessels on inland waterways. Both criteria, however, are directly affected by the vessels' sailing speed. In this paper, we present a mathematical programming formulation of the speed optimization problem, which aims at minimizing the aggregated fuel consumption on an inland waterway network. The network can consist of multiple river segments, connected by a set of locks, without restrictions on the configuration. To allow scalability towards realistic waterway networks, we also propose a local-search based heuristic to optimize the speed for individual vessels. We evaluate the effectiveness of the heuristic by comparing it to solving the exact mathematical programming formulation. For all computational experiments, we make use of real AIS data from a section of the Dutch river network. We observe that the heuristic is able to construct a high quality solution in realistic problem settings within reasonable amount of computation time.

1. Introduction

The transportation sector plays a vital role in today's globalised economical system. However, the sector is challenged by the growing need for economic, social and environmental sustainability. In light of the long term sustainability goals imposed by Europe, the European Commission has called for a reduction by 2030 of at least 20% in greenhouse gas emissions within the transportation sector compared to their 2008 level. To achieve this ambitious goal, the sector should invest in a shift towards more sustainable transport modes, as well as implementations of new technologies for traffic management to lower transport emissions, see [7].

Despite all efforts over the past decade(s), the majority of transport operations are still performed on the road using trucks, see [8]. In comparison to road transportation, the use of inland waterways is more environmentally friendly because of lower greenhouse gas emissions per volume or weight unit, it is relatively cheap due to economies of scale and the increased bundling opportunities. However, the inland waterway network is less dense than the road network.

Additionally, it contains many locks to overcome height differences between two adjacent river segments. As the capacity of these locks is limited, they form bottlenecks along the river network. Lack of coordination between individual skippers, i.e., the persons in charge of vessels, is one of the major factors to the formation of queues. To compensate for the waiting time, skippers need to speed up after a lock to arrive on time at their destination. Moreover, the lock management, i.e., the scheduling of all lock operations, is done based on intuition and ad-hoc decision making, typically resulting in a first-come-first-served queuing policy.[1] To ensure timely service at the lock, the skippers have the incentive to speed up in front of a lock, and overtake preceding vessels to receive the lock service earlier.

The fuel consumption of a vessel is largely dependent on the required power, and therefore also on the speed of the vessel. As shown above, the presence of locks and their way of operating results in racing behaviour of individual skippers before the lock and after the lock to arrive at their destination in due-time. Equipped with adequate algorithms, a digitalization of the management of waterways can lead to increase in coordination and therefore to reduction in fuel consumption and emission.

☆ This manuscript was processed by Associate Editor Furini.

* Corresponding author at: Hamburg University of Technology, Institute for Algorithms and Complexity, Hamburg, Germany.
E-mail address: julian.golak@tuhh.de (J.A.P. Golak).

[1] Communicated to us by our industrial partner Trapps Wise. B.V. (https://trappswise.nl/).

https://doi.org/10.1016/j.omega.2021.102580

In this paper, we introduce an algorithm to coordinate a fleet of vessels on an inland waterway network. This algorithm can be extended to a decision support tool and can be used by lock operators to schedule the lock operations of arriving ships, and will provide skippers with speed advice to minimize the overall fuel consumption on the river, and therefore increase the efficiency of inland waterway transportation. As the algorithm relies on a powerful local-search based heuristic, it is able to schedule instances of realistic size in a very reasonable time.

Nowadays, it is obligatory for vessels to be equipped with AIS tracking system. These transmitters send out live data, e.g., GPS, destination, speed, and global attributes of the vessel, e.g., vessel ID, vessel type, size. This means that all required information is automatically made available by the users of the inland waterway. Route specific characteristics, such as arrival/destination time and location is required from and provided by the skippers.

The results of this paper can be used by policymakers to evaluate the attractiveness of imposing regulations on skippers in inland waterways by straightforward comparison of total fuel consumption and can help to decrease greenhouse gas emissions.

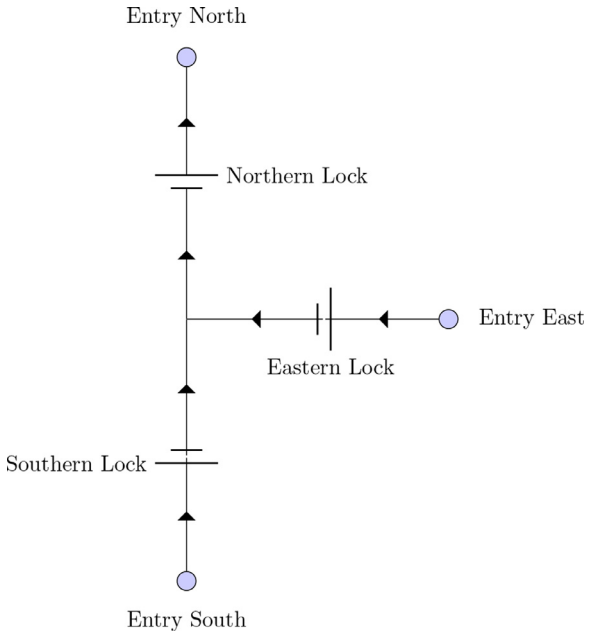

Fig. 1. A River Network.

2. Literature review

2.1. Lock scheduling

Due to its practical importance optimization of inland waterways has been the focus of many publications. These publications focus on different variants of the lock scheduling problem. The scheduling problem of minimising total waiting time subject to the natural restrictions of a single lock with given arrival times is investigated by Passchyn et al. [15] and Passchyn et al. [14]. The authors provide a polynomial time algorithm for a lock with identical chambers and discuss the complexity of generalisations. In [21] and [22], the problem of positioning a set of vessels into the lock chambers, while satisfying a number of general and specific placement constraints, is discussed.

These results are generalised for a sequence of locks by [9–11,13,17]. In [13], a mathematical programming formulation for optimizing the schedule with respect to total flow time or total emission is presented. This model is restrictive, as it does not allow for multi-chambered locks and it assumes that all vessels require to travel through the entire passage. Prandtstetter et al. [17] introduce a heuristic to minimise the overall vessel travel times and test it against real-world vessel trajectories on the Danube River. The authors in [9–11] consider the problems of scheduling locks under general assumptions while minimizing the total flow time of the fleet.

There are more publications improving efficiency of lock sequences on specific, existing river sections. Petersen and Taylor [16] enhance the lock scheduling on the Welland Canal in North America. The Upper Mississippi River in the US is used as a case study in [18–20].

To our best knowledge, only [5,6,13] minimise overall greenhouse gas emissions by optimising the speed at which vessels approach the locks. In [6] the problem is modeled in a game theoretic setting. The optimisation of speeds under uncertainty has been addressed in [5]. In [13] authors proposed a mathematical programming formulation for a sequence of locks, which is closely related to the scope of the article at hand.

More publications have been focused on more general inland waterway ship routing and port management problems, see e.g. [1–4,12,23–25].

2.2. Our contribution

Unfortunately, the mathematical programming formulation proposed in [13] is of limited practical use. This is because the large-scale instances coming from the real inland waterways quickly become intractable. In this paper, we aim to tackle the large-scale cases and we extend and generalise the results of [13] in several ways. First, we allow for a network of waterways with multiple entry and exit points, and any lock configuration. Second, vessels have individual arrival and destination locations in the network. These locations can be entry/exit points of the system or any location along the waterway. This gives skippers an opportunity to choose ports or anchor points as destinations. Finally, our methods also allow for multi-chambered locks. Most importantly, we design a lock-search based heuristic to tackle real-life large-scale instances of the lock scheduling problem minimising the total greenhouse gas emission of the fleet of vessels.

The paper is structured as follows. In Section 3, we define the problem and present its mathematical programming formulation. In Section 4, a heuristic is constructed that first finds an initial schedule and then iteratively applies a local-search operator to improve the schedule. In Section 5, we report on numerical experiments comparing the quality, running time and robustness of the constructed heuristic against the performances of a conventional solver dealing with the mathematical programming formulation. Finally, in Section 6 we give a concluding discussion and ideas for future research.

3. The speed optimization problem

3.1. Mathematical notation

Consider an inland waterway network that consists of a set of river segments with given lengths. These segments connect a set of locks and multiple entry/exit points. We refer to this set of locks as $\mathcal{L}$. Note that, in contrast to existing literature, we do not assume that locks are sequentially placed along one river segment, but that any lock configuration is feasible. An example of such a river network is depicted in Fig. 1. In this figure, the lines represent river segments and the arrows point in the downstream direction. The

circles depict the entry/exit points of the river system and the parallel lines show the locks.

Over a time period from 0 to *TIME_MAX*, vessels arrive, navigate through the network and leave the system. Let $\mathcal{S}$ define the set of vessels. For each vessel $s \in \mathcal{S}$, the time of arrival into and the time of desired departure out of the system are given and denoted by A_s and D_s, respectively. Vessels may enter and exit the system at entry/exit points, or skippers can anchor their vessel at any location on a river. In the latter case, the arrival/exit location is between the two endpoints of that specific river segment. The trajectory of a vessel is the shortest path from arrival location to destination. Let the sequence of locks a vessel $s \in \mathcal{S}$ needs to cross be given by $\mathcal{L}_s = \{\ell_{s,1}, \ldots, \ell_{s,n_s}\}$ where $\mathcal{L}_s \subseteq \mathcal{L}$. The lengths of the river segments that vessel s needs to cross is defined as $\mathcal{D}_s = \{d_{s,1}, \ldots, d_{s,n_s+1}\}$. In case a skipper starts or ends by anchoring inside the system, the first or the last segment is only partly crossed. Thus, $d_{s,1}$ and d_{s,n_s+1} may only be a fraction of the length of the river segment. The other distances, i.e. between two locks, are the complete lengths of the river segment. Finally, a vessel can approach a lock coming from up or downstream. Let $\mathcal{W}_s = \{w_{s,1}, \ldots, w_{s,n_s}\}$ be the set of directions, where $w_{s,i} \in \{up, down\}$ is the direction from which vessel s approaches its ith lock. Note that, for any vessel, the directions can alternate for each lock in its path. An example is a vessel which would travel from Entry South to Entry East in Fig. 1. In such a case, the vessel would approach the Southern Lock from downstream and the Eastern Lock from upstream.

It might happen that the ports of call on route are located between the locks, e.g., between two locks a vessel requires to pick up shipments at a port and then to continue. In this case, without loss of generality, we assume that the route of the vessel has to be subdivided and communicated to the solver/decision maker as two (or more) distinct routes. The first route is from the start location to the port and the second is from the port to the destination. The vessel has a scheduled time at each port and anchor point, so, every skipper knows exactly her departure and arrival times for all origins and destinations. Thus, this subdivision operation is perfectly feasible in practice.

Let E_s denote the *lock size* of a ship $s \in \mathcal{S}$. This lock size could be a simple counter (i.e., every ship has size one) or the area that the ship occupies in a lock. A lock may consist of multiple independently working chambers. The number of chambers for each lock $\ell \in \mathcal{L}$ is denoted by B_ℓ. We assume that all chambers of a lock are identical. For each lock $\ell \in \mathcal{L}$, the capacity of a chamber, i.e., the total size of vessels that can be accommodated in a chamber, is given by C_ℓ. The processing time, i.e., the time it takes to load, unload and process a batch of vessels in a chamber, is defined by T_ℓ. We assume that every batch has a homogeneous processing time. Throughout the paper, we disregard the shapes of vessels and we treat the sizes of vessels and the capacities of chambers in this very simplified way to avoid solving a hard problem of packing vessels in the lock, otherwise it leads to much higher running time for both, the exact solver and the local-search heuristic.

Example. Let us clarify some definitions, using the structure of the Amsterdam Rhine Canal case, which is introduced more elaborately in Section 5.1. Here, the set of locks is defined as $\mathcal{L} = \{\text{north}, \text{east}, \text{south}\}$. The setup is illustrated in Fig. 1. Consider a vessel s that enters the system at the south entry and exits at the northern entry. Thus, it first sails downstream from entry south to the southern lock. Then it continues to sail downstream from the southern lock to the northern lock and finally leaves the system at Entry North. Formally, $\mathcal{L}_s = \{\text{south}, \text{north}\}$ and $\mathcal{W}_s = \{down, down\}$. The distances on each segments are respectively $\mathcal{D}_s = (9.39, 16.20, 18.84)$. Additionally, assume that this vessel s arrives into the system at 7:31 AM and desires to leave it at 12:31 AM. Throughout the entire article, we use minutes after midnight as the time unit. This translates to an arrival times of $A_s = 451$ and a departure time of $D_s = 751$. The processing time of a batch at Northern Lock and Southern Lock are 22 and 23 min, respectively, i.e., $T_{north} = 23$ and $T_{south} = 22$. Under the assumption that there are no waiting times at the locks, the vessel has, therefore, a sailing time of 255 min, or 4 h and 15 min ($D_s - A_s - T_{north} - T_{south}$).

The fuel consumption of each vessel is derived as in [5]. The resistance force for a vessel is proportional to the square of its velocity, i.e., v^2. The fuel consumption per unit of time is proportional to the required power, which brings to the resistance an additional multiplicative factor of v. Thus, the fuel consumption per unit of time is cubic in the velocity of the vessel, i.e., v^3. The required time for sailing distance d at constant speed v is d/v. Therefore, for a vessel $s \in \mathcal{S}$ sailing distance d at constant speed v, the total fuel consumption equals $\alpha_s d v^2$, where the constant α_s is vessel dependent. Then, the cost for a skipper is fully characterised by the fuel consumption multiplied by the per-unit cost of fuel. This is in line with conventions from related literature, see [13].

Let $v_{s,i}$ denote the speed of vessel s on the ith segment of the waterway network, and let $\mathcal{V}_s = (v_{s,1} \ldots, v_{s,n_s+1})$. Then the fuel consumption of vessel $s \in \mathcal{S}$ is

$$C_s(\mathcal{V}_s) = \sum_{i=1}^{n_s+1} \alpha_s d_{s,i} v_{s,i}^2, \tag{1}$$

and the total fuel consumption of the fleet in a solution is given by

$$C_{total}(\mathcal{V}) = \sum_{s \in \mathcal{S}} C_s(\mathcal{V}_s), \tag{2}$$

which is the expression that we aim to minimize.

The solution of this optimization problem is a schedule of lockage starting times (i.e., the time at which the service of a vessel at the lock starts) at each chamber of each lock, an assignment of vessels to specific lockages, and a speed for each vessel on each river segment between origin and destination. The schedule is feasible if all of the following conditions are satisfied:

- Each vessel must pass all locks along its itinerary in the given order and from the given directions.
- The speed of each vessel must be consistent with the lockage starting times, the arrival time, and the deadline.
- The speed of each vessel should be in the interval $[0, V_s^{\max}]$ at all times, where $V_s^{\max}$ represents the maximum speed at which vessel s can sail. We assume that this maximum speed does not exceed the speed limit of the river segments in the network. We assume that there is no minimum speed throughout the paper.
- The lockages of each lock chamber must alternate between downstream and upstream, and every pair of lockages is at least T_l time units apart.
- The total size of vessels assigned to one lockage/batch must not exceed the capacity of the lock chamber.

Example. Let us again consider vessel s on the Amsterdam Rhine Canal as illustrated in Fig. 1. Let $\mathcal{L}_s = \{\text{south}, \text{north}\}$, $\mathcal{W}_s = \{down, down\}$ and $\mathcal{D}_s = (9.39, 16.20, 18.84)$. Furthermore, assume that $A_s = 451$, $D_s = 751$ and $C(v) = v^2$. Also note that the processing time of a batch at Northern Lock and Southern Lock are 22 and 23 min, respectively, i.e. $T_{north} = 23$ and $T_{south} = 22$.

Vessel s chooses to sail at a constant speed of 0.3 km per minute on each river section, that is $v_s = (0.3, 0.3, 0.3)$. For reasons of simplicity, we assume that the lockages are fully synchronized with the speed of vessel s, meaning that vessel s does not have to wait in front of the locks. The arrival time of vessel s at

its first lock is then $A_s + d_{s,1}/v_{s,1} = 482.3$ min. Similarly, the arrival time of vessel s at the second lock is $482.3 + T_{l_{s,1}} + d_{s,2}/v_{s,2} = 558.3$ min. Lastly, the vessel arrives at the destination at time $558.3 + T_{l_{s,2}} + d_{s,3}/v_{s,3} = 644.1$ min. Conclusively, vessel s arrives at its destination before its specified deadline. Assuming for simplicity that $\alpha_s = 1$, the fuel consumption of vessel s is then calculated as follows:

$$C_s(\mathcal{V}_s) = d_{s,1}v_{s,1}^2 + d_{s,2}v_{s,2}^2 + d_{s,3}v_{s,3}^2 = 3.99 \text{ units of fuel} \tag{3}$$

3.2. Mixed-Integer linear program

In this section, we describe a Mixed-Integer Linear Program (MILP) formulation to compute an optimal speed for every vessel on each river section as this solution minimizes total aggregated fuel consumption. The MILP is an extension of the model from Passchyn et al. [13], which aimed at minimizing emissions on a waterway with multiple locks. Our MILP differs from the model by Passchyn et al. [13] in three ways:

1. The setup is extended to a network of waterways in which locks are placed in any configuration.
2. Vessels are allowed to have individual arrival and destination locations in the network.
3. Locks can have multiple chambers.

Sets The decision variables in the model are based on possible lockages. Assuming an adversary lock operator, each vessel would face the lock position to its opposite side. Therefore, the number of lockages in the optimal solution does not need to be greater than double the number of ships. Thus, the upper bound for the number of lockages is given by $K = 2|S|$. The set of possible lockages is, therefore, defined as $\mathcal{K} = \{1, \dots, K\}$. For each lock $\ell \in \mathcal{L}$, $\mathcal{S}_\ell^{up}$ and $\mathcal{S}_\ell^{down}$ represent the set of vessels that approach lock ℓ form upstream and downstream, respectively.

Decision Variables Let $v_{s,i}$ be a decision variables for speed of vessel $s \in \mathcal{S}$ on segment i of the waterway network. Let $\delta(\ell, j, k)$ be the starting time of the k'th lockage of the j'th chamber of lock $\ell \in \mathcal{L}$ for all $k \in \mathcal{K}$ and $j \in \{1, \dots, B_\ell\}$. Furthermore, for each $s \in \mathcal{S}$, $\ell \in \mathcal{L}_s$, $j \in \{1, \dots, B_\ell\}$ and $k \in \mathcal{K}$, we define $z(s, \ell, j, k)$ as follows:

$$z(s, \ell, j, k) = \begin{cases} 1, & \text{if vessel } s \text{ is handled by the} \\ & k\text{th lock movement of chamber } j \text{ of lock } \ell. \\ 0, & \text{otherwise} \end{cases}$$

MILP formulation

$$\min \sum_{s \in S} \sum_{i=1}^{n_s} \alpha_s d_{s,i} v_{s,i}^2 \tag{4}$$

Subject to

$$A_s \leq \delta(\ell_{s,1}, j, k) - d_{s,1}/v_{s,1} + M_s^a(1 - \sum_{\kappa=0}^{k} z(s, \ell_{s,1}, j, \kappa)) \forall s, j, k \tag{5}$$

$$D_s \geq \delta(\ell_{s,n_s}, j, k) + T_{\ell_{s,n_s}} - d_{s,n_s+1}/v_{s,n_s+1} - M_s^d(1 - \sum_{\kappa=k}^{K} z(s, \ell_{s,n_s}, j, \kappa)) \quad \forall s, j, k \tag{6}$$

$$z(s_1, \ell, j, k) + z(s_2, \ell, j, k) \leq 1 \forall \ell, j, k, s_1 \in \mathcal{S}_\ell^{up}, s_2 \in \mathcal{S}_\ell^{down} \tag{7}$$

$$z(s_1, \ell, j, k-1) + z(s_2, \ell, j, k) \leq 1 \forall \ell, j, k > 1, s_1 \in \mathcal{S}_\ell^{up}, s_2 \in \mathcal{S}_\ell^{up} \tag{8}$$

$$z(s_1, \ell, j, k-1) + z(s_2, \ell, j, k) \leq 1 \quad \forall \ell, j, k > 1, s_1 \in \mathcal{S}_\ell^{down}, s_2 \in \mathcal{S}_\ell^{down} \tag{9}$$

$$\delta(\ell, j, k) - \delta(\ell, j, k-1) \leq T_\ell \quad \forall \ell, j, k > 1 \tag{10}$$

$$\sum_{s \in \mathcal{S}} E_s z(s, \ell, j, k) \leq C_l \quad \forall \ell, j, k \tag{11}$$

$$\sum_{j=0}^{B_\ell} \sum_{k \in \mathcal{K}} z(s, \ell, j, k) = 1 \quad \forall s, \ell \tag{12}$$

$$\delta(\ell_{s,i-1}, j, k) \geq \delta(\ell_{s,i}, j, k) + T_{\ell_{s,i}} + v_{s,i}/d_{s,i} - M_{k_1,k_2}(2 - \sum_{\kappa_1=1}^{k_1} z(s, \ell, j, \kappa_1) - \sum_{\kappa_2=k_2}^{K} z(s, \ell, j, \kappa_2)) \quad \forall s, 1 < i \leq n_s, j, \ k_1 < k_2 \tag{13}$$

$$0 \leq v_{s,i} \leq V_s^{\max} \quad \forall s, i \tag{14}$$

$$\delta(\ell, j, k) \geq 0 \quad \forall \ell, j, k \tag{15}$$

$$z(s, \ell, j, k) \in \{0, 1\} \quad \forall s, \ell, j, k \tag{16}$$

The objective function (4) minimizes the total fuel consumption. Constraint (5) ensures that vessels arrive at the lock before their respective lockage time has started and constraints (6) ensure that vessels are leaving the system before their deadline. Notice that both constraints should hold for all possible assignments of vessel s to any lockage k of any chamber j of the first and the last lock of vessel s, respectively. Constraints (5) and (6) are based on values M_s^a and M_s^d, which are large constants specified in [13]. Constraints (7) to (13) are used to model the working of the lockages. Thus, (7) ensures that lockages are alternating between upstream and downstream. Constraints (8) and (9) ensure that only vessels from one side can enter a lockage. The processing time of a lockage is modeled by Constraint (10), while the capacity of a chamber is modeled by (11). Constraints (12) and (13) ensure that each vessel is processed by all of its locks and that it arrives at each lock before its assigned lockage time. Constraints (13) make use of large constants M_{k_1,k_2} as specified in [13]. Compared to the MILP formulation of [13], we have broadened the set notation such that each vessel has an individual sequence of locks to pass and therefore, the travelled distances are also vessel specific.

4. Local search heuristic

Due to the limited scalability of the MILP, it would be impossible to solve realistic instances within an acceptable computation time. To allow our methods to be used in practice, we propose a heuristic to determine high-quality speed advice for all skippers on the waterway network within a reasonable amount of computation time.

Table 1
Schedule of starting times.

Eastern Lock			Northern Lock			Southern Lock		
k	*up*	*down*	*k*	*up*	*down*	*k*	*up*	*down*
0	0	22	0	0	23	0	0	22
1	44	66	1	23	46	1	22	44
2	88	110	2	46	69	2	44	66
3	132	152	3	69	92	3	66	88
...	...	...	...	...	...	...	...	...
31	1364	1386	61	1403	1426	64	1408	1430
32	1408	/	62	1426	/	65	1430	/

4.1. Preliminaries and assumptions

To reduce the solution space and, consequently, the complexity of the speed optimization problem, we impose a fixed schedule on the starting times of lockages. This is in contrast to the MILP formulation, where the lockage could be scheduled at any feasible time throughout the planning horizon. For each chamber independently, lockages are set consecutively, such that there is no idle time between two lockages and that they are alternating between up and downstream. Furthermore, the starting times of the first lockage of each chamber is evenly spread on the interval $[0, 2T_l]$. As a result, each lock in each direction has a chamber starting the lockage process every $T_l^* = 2T_l/B_l$ time units.

Formally, let $K_l = \lfloor TIME\_MAX \rfloor / T_l^*$ represent the maximum number of lockages per direction for lock l. Define $t(l,k,w)$ as the starting time of the kth lockage in direction w of lock l. Then, fix the starting times in the following way:

$$t(l,k,up) = 2\ k\ T_l^* \qquad \forall l \in \mathcal{L}, k = \{0,\dots,K_l/2\}$$
$$t(l,k,down) = 2\ (k+1)\ T_l^* \qquad \forall l \in \mathcal{L}, k = \{0,\dots,K_l/2\}$$

Similarly, let $h(l,k,w)$ represent the set of vessels assigned to the kth lockage, in direction w of lock l. In the initialisation, every element of that set is declared empty, i.e. $h(l,k,w) = \emptyset$ for $l \in \mathcal{L},\ k = 0,\dots,K_l,\ w \in \{up, down\}$.

Example. We apply the initialisation procedure on the Amsterdam Rhine Canal case. As data is expressed in minutes, *TIME_MAX* is set to 1440. Given the measured parameters for this case, summarised in Table 4, the time between two lockages and the maximum number of lockages for a lock l, i.e. T_l^* and K_l are:

$$T^*_{east} = \frac{2T_{east}}{B_{east}} = 44,\ K_{east} = \lfloor \frac{TIME\_MAX}{T_{east}} \rfloor = 32$$

$$T^*_{north} = \frac{2T_{north}}{B_{north}} = 23,\ K_{north} = \lfloor \frac{TIME\_MAX}{T^*_{north}} \rfloor = 62$$

$$T^*_{south} = \frac{2T_{south}}{B_{south}} = 22,\ K_{south} = \lfloor \frac{TIME\_MAX}{T^*_{south}} \rfloor = 65$$

The resulting schedule of lockages is given in Table 1.

The arrival times of vessels at every lock are determined by their speed assignment. Assume that a vessel is assigned to lockage k' of lock l' going in direction w'. If the arrival time of that vessel is later than $t(l',k',w')$, the schedule becomes infeasible as the assigned lockage is inconsistent with the assigned speed. On the other hand, if the arrival time of that vessel is earlier than $t(l',k',w')$, then the vessel has idle time at that lock. This implies that the speed of vessel s can be reduced on the river segment before lock l'. Thus, the total fuel consumption is reduced while not altering the arrival times at any other lock. Consequently, in an optimal schedule, the arrival time at a lock is synchronised with the starting time of the assigned lockage at that lock. This shows, that the assignment of lockages implies the assignment of speeds for a given vessel and vice versa.

The speed assignment on the last segment is equivalent to the distance divided by remaining time on that segment. This means that the optimal speed on this river segment is induced by the speeds on the other river segments or, formally, that

$$v_{s,n_s+1} = \min\left\{ \frac{d_{s,n_s+1}}{\tilde{T}_s - \sum_{i=1}^{n_s} d_{s,i}/v_{s,i}}, V_s^{\max} \right\} \tag{17}$$

where $\tilde{T}_s = D_s - A_s - \sum_{l=1}^{n_s} T_l$.

4.2. Construction of the initial solution

Given the predefined schedule of lockage starting times, vessels are assigned to spots in the following manner. First, for each vessel s, let $\tilde{\mathcal{L}}_s$ represent the set of locks, which have not processed vessel s, yet. Initialise $\tilde{\mathcal{L}}_s = \mathcal{L}_s$ for all s. Second, vessels are iteratively selected according to a *least slack time* priority rule until all vessels are scheduled. Each time a vessel s is selected, it is scheduled to the first lock in $\mathcal{L}_s$, say $l_{s,i'}$. On $l_{s,i'}$, the vessel is *assigned to the earliest feasible lockage* that it can reach. Given the speeds, assigned on all segments before i', $v^0_{s,1},\dots,v^0_{s,i'-1}$, this earliest reachable lockage is defined by

$$k^0_{s,i'} = \tau_{s,i'}(v^0_{s,1},\dots,v^0_{s,i'-1}, v^{\max}) \tag{18}$$

The following two scenarios might occur:

1. $\sum_{s' \in h(l_{s,i'},k^0_{s,i},w_{s,i'})} E_{s'} + E_s > C_{l_{s,i'}}$
   This means that $k^0_{s,i'}$ is not feasible with respect to the lock capacity. If this is the case, then k^0 is incremented by one until a feasible lockage is attained.
2. $\sum_{s' \in h(l_{s,i'},k^0_{s,i},w_{s,i'})} E_{s'} + E_s \le C_{l_{s,i'}}$
   This means that k^0 is feasible. If this is the case, s is added to the set of the respective lockage. Furthermore, the lockage assignment is updated by adding $k^0_{s,i'}$ to $\mathcal{K}_s$ and the speed assignments are updated by setting $v^0_{s,i'} = \tau^{-1}_{s,i'}((k^0_{s,j})_{j\le i})$. Finally, the set of locks that have not yet processed s is updated by removing $l_{s,i'}$ from $\tilde{\mathcal{L}}_s$. A vessel is labeled as scheduled when $\tilde{\mathcal{L}}_s$ is empty. The speed on the last segment is then computed by Eq. (17).

For a vessel s, let the resulting speeds be denoted by $v^0_s = (v^0_{s,1}\dots,v^0_{s,n_s+1})$. Similarly, let $\mathcal{K}^0_s = \{k^0_{s,1},\dots,k^0_{s,n_s}\}$, where $k^0_{s,i}$ is the assigned lockage of vessel s at its ith lock.

Example. We continue our example of the Amsterdam Rhine Canal. Consider again vessel s with following specifications:

$$\tilde{\mathcal{L}}_s = \mathcal{L}_s = \{\text{south}, \text{north}\}$$
$$\mathcal{W}_s = \{down, down\}$$
$$\mathcal{D}_s = (9.39, 16.20, 18.84)$$

$$A_s = 451$$
$$D_s = 751$$
$$C(v) = v^2 \ (\text{with } v_{\max} = 0.41)$$

Assume vessel s is selected, the process of assigning s to lockages is visualised in Figs. 2a to d. In Fig. 2a, time is plotted on the vertical axis and travel distance is plotted on the horizontal axis. Initially, vessel s is at distance 0 at time $a_s = 451$ and is not yet scheduled on any lock. From this starting point all lockages on $l_{s,1}$ in the green region can be reached, which is constrained by the speed limit. At maximum speed, vessel s arrives at the first lock at time $a_s + d_{s,1}/v_{\max} = 473.90$. Given the predetermined schedule,

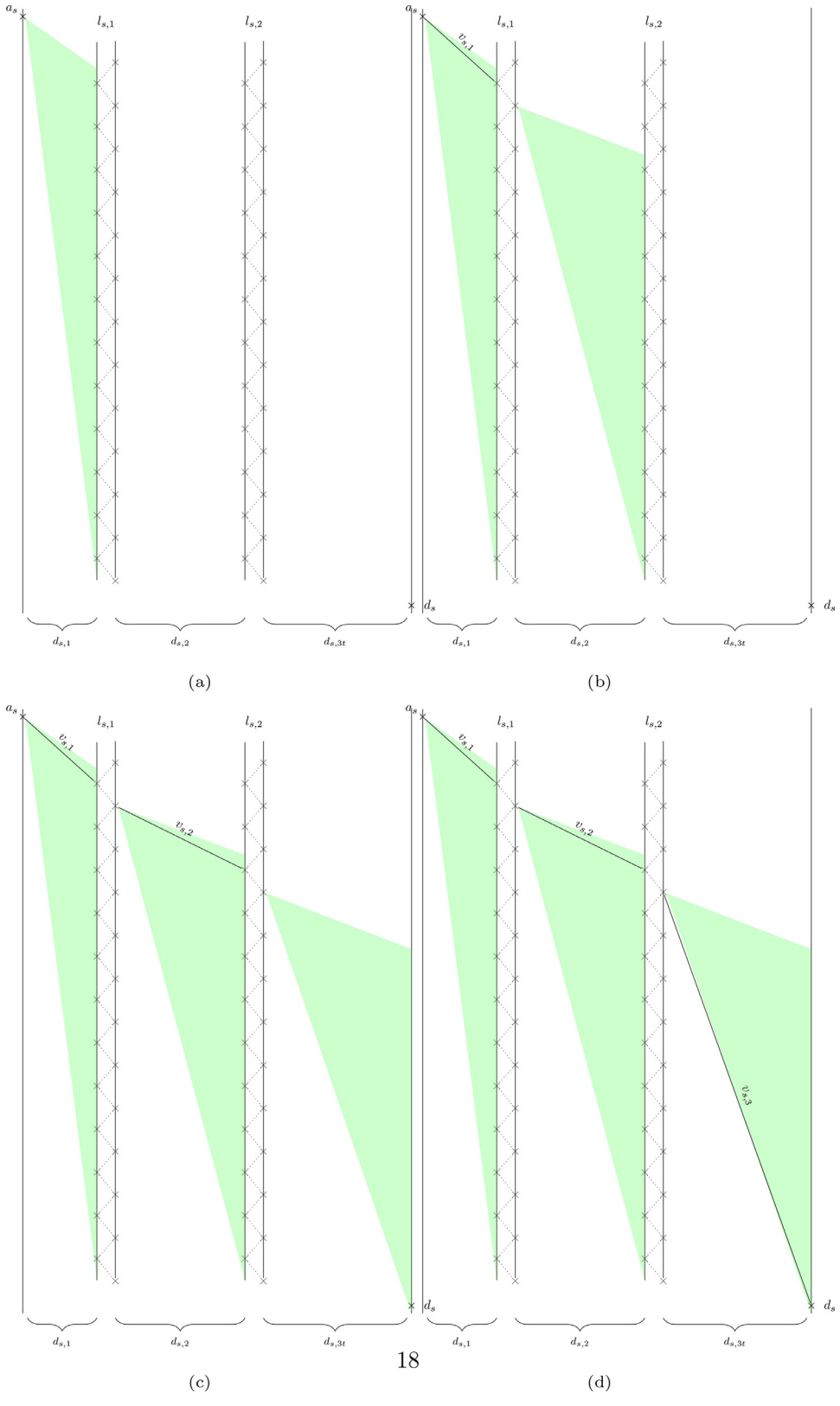

Fig. 2. Different steps to construct an initial solution for each vessel.

the next lockage of the southern lock processing vessels coming from downstream is $k = 21$ with $t(south, down, 1) = 484$. Assuming that the lockage has capacity left, vessel s is assigned to it. Furthermore, the lock is removed from $\bar{\mathcal{L}}_s$ and s is assigned to the lockage, i.e. $k^0_{s,1} = 21$. The speed on the first segment is synchronized to the starting time of the lockage, i.e. $v_{s,1} = 0.284$.

When assigned to lockage $k = 21$, vessel s is processed by the southern lock at time $t = 506$ and is now scheduled on the northern lock as shown in Fig. 2b. Traveling at maximum speed would

result in an arrival time of $506 + d_{s,2}/v_{\max} = 545.51$, and the next available lockage is $k = 23$ with $t(south, down, 4) = 552$. Assuming there is capacity left, the vessel is assigned to the lockage. Similarly, the northern lock is removed from $\tilde{\mathcal{L}}_s$, $k^0_{s,2} = 23$ and $v_{s,2} = 0.352$. Since $\tilde{\mathcal{L}}_s$ is empty, the vessel is labeled as scheduled.

The speed from northern lock to exit point can be computed by Eq. (17). The final assignment of speeds and lockages are then given by $\mathcal{K}^0_s = \{21, 23\}$ and $v^0_s = (0.285, 0.352, 0.107)$ and visualised in Fig. 2d. The fuel consumption of vessel s under the initial schedule is 2.9871.

Since the vessels are scheduled on the earliest feasible locks available to them, the schedule resulting from the initial solution is costly. By construction, high speeds are advised on earlier segments while skippers can maintain a very low speed on the last segment. Due to convexity of the fuel consumption function, this unbalanced speed assignment results in a high total cost. Although this might seem inconvenient, the proposed procedure performs very well for generating a feasible schedule. The great dependency between decisions for different time intervals and at different locations in the network have forced us to prioritize feasibility over solution quality. Due to high complexity of the system, fixing infeasible operations locally will likely create multiple new conflicts in the schedule.

To further improve the quality of our solution, we propose a dedicated local search operator, which focuses on rebalancing again the speed assignments.

4.3. Balancing speeds using a gradient descent procedure

As discussed above, the required speed on the final segment of each vessel is deduced directly from its speed on all previous segments. Therefore, we will focus solely on adjusting explicitly the speeds on segments 1 to n_s.

First, we introduce the gradient of the cost function.

$$\begin{aligned} C_s(v_s) &= \sum_{i=1}^{n_s} d_{s,i} v_{s,i}^2 + d_{s,n+1}\left(\frac{d_{s,n_s+1}}{\tilde{T}_s - \sum_{i=1}^{n_s} d_{s,i}/v_{s,i}}\right)^2 \\ \frac{\partial C_s}{\partial v_{s,i}}(v_s) &= 2 d_{s,i} v_{s,i} - \frac{2 d_{s,i} d_{s,n_s+1}^3}{(\tilde{T}_s - \sum_{i=1}^{n_s} d_{s,i}/v_{s,i})^3 v_{s,i}^2} \\ \nabla C_s(v_s) &= \left(\frac{\partial C_s}{\partial v_{s,i}}\right)_{i=1,\dots,n_s} \end{aligned} \tag{19}$$

The gradient of the cost function at v^0_s (i.e., the speed vector of the initial solution) gives the direction of steepest increase in cost. Consequently, to attain the largest decrease in cost, the speeds of vessel s need to be shifted according to $v_s(\gamma) = v^0_s - \gamma \nabla C_s(v^0_s)$, where γ denotes the magnitude of the shift. For each value of γ there exists a corresponding lockage assignment, which is defined as the latest possible lockage reachable given the adjusted speeds.

Let Γ be the candidate set of unique and feasible lockage combinations for all values of $\gamma \geq 0$. The construction of the candidate lockage assignments Γ is done as follows. For each vessel s, we define $\Delta k_i \in \mathbb{N}$ as the shift of lockages at lock $l_{s,i}$, and $\hat{\gamma}$ as the magnitude of gradient step required for that shift. In other words, $\Delta k_i \in \mathbb{N}$ denotes by how many lock iterations the service of vessels will be advanced (negative Δk_i) or postponed (positive Δk_i). As all lockages were predefined and fixed, we can rewrite $\hat{\gamma}$ as follows:

$$\begin{aligned} \frac{d_{s,i}}{v^0_{s,i} - \hat{\gamma}_i \nabla C_{s,i}} &= \frac{d_{s,i}}{v^0_{s,i}} + 2\Delta k_i T^*_{l_{s,i}} \\ \hat{\gamma}_i &= \frac{(v^0_{s,i})^2 \Delta k_i T^*_{l_{s,i}}}{\nabla C_{s,i}(d_{s,i} + v^0_{s,i} \Delta k_i T^*_{l_{s,i}})}. \end{aligned} \tag{20}$$

The candidate set is now constructed by considering different values for Δk_i. First, we initialise $\Delta k_i = 0$ for all i. Second, Δk_i is incremented (if the gradient is positive) or decremented (if the gradient is negative) by one. The index with the lowest $\hat{\gamma}_i$ is chosen to become permanent, while the other indices are set back to their previous values. $K^c = (K^0_i + \Delta k_i)_{i=1,\dots,n_s}$ is added to the set of candidate solutions and the process is repeated. Once a lockage corresponds to an infeasible speed assignment (e.g., a negative speed, or a speed above $v_{\max}$) or the resulting lockage does not exist in the schedule, the iteration stops.

Table 2
List of candidate lockage assignments.

	Lockage Assignment	$\hat{\gamma}$	Cost
K^c_1	{21, 24}	0.054	1.9397
K^c_2	{21, 25}	0.080	1.6583
K^c_3	{21, 26}	0.096	1.6649
K^c_4	{21, 27}	0.107	1.9281
K^c_5	{22, 27}	0.113	**1.5337**
K^c_6	{22, 28}	0.115	2.2802
...	...	...	

Example. We continue our example. Consider again vessel s with initial lockage assignment $\mathcal{K}^0_s = \{21, 23\}$ and speed assignments $v^0_s = (0.285, 0.352, 0.107)$. The balancing procedure is visualised in Fig. 3.

First, the gradient is computed using Eq. (19), which results in $\nabla C_s(v_s) = (1.0, 2.2)$. We find the candidate assignments given in Table 2. From these candidate assignments, the feasible lockage assignment with minimum cost is $\{22, 27\}$. Thus, vessel s is reassigned accordingly. The new cost of this assignment is 1.5337, while the cost of the initial assignment was 2.9871.

5. Computational experiments

5.1. Introduction to the amsterdam rhine canal case

To evaluate the performance of our heuristic, we perform computational experiments on a dataset extracted from an inland waterway network, located at the split of Rhine and the Amsterdam-Rhine Canal (The Netherlands). The network contains three locks, connected by river segments as visualized in Fig. 4.

The river network connects the *Rhine*, which flows from East to West, with the *Amsterdam Rhine Canal*, which flows from South to North. Furthermore, the Amsterdam Rhine Canal intersects with both the Rhine, and the *Nederrijn*, which flows parallel to and north of the Rhine.

There are three locks:

- The *Prinses Irenesluizen* (referred to as *Northern Lock*). This lock consists of two separate chambers that process vessels independently. The distance from the northern entry point of the river system to the Northern Lock is 22.13 km.
- The *Prins Bernhardsluizen* (referred to as *Eastern* Lock). This lock is used far less than the other two locks and thus consists only out of one chamber. The distance from the Eastern to the Northern Lock is 10.22 km, while the distance from the Eastern Lock to the Eastern exit point is 19.576 km.
- The *Prinses Marijkesluizen* (referred to as *Southern Lock*) forms the connection point between the Rhine and the Amsterdam Rhine Canal. This lock consists of two independent chambers. The distances from Southern lock to Eastern and to Northern Locks are 23.11 km and 16.84 km, respectively. Furthermore, the western and eastern entry points into the river system are located at 12.936 km and 13.896 km, respectively.

5.2. Preprocessing of the dataset

The heuristic was tested on real-life scenarios extracted from the Automatic Identification System (AIS). We make use of real-life AIS data from the period 2 January 2019 – 31 March 2019, split

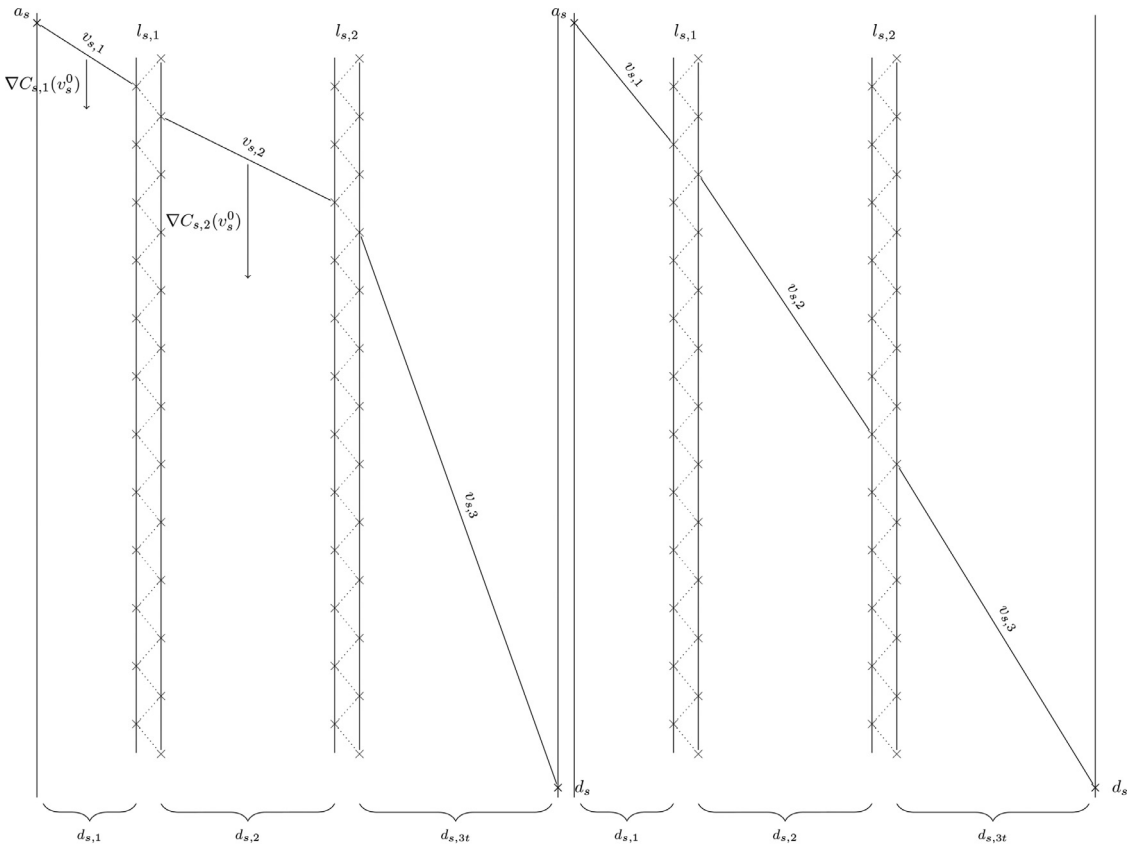

Fig. 3. Balancing Speeds using a gradient descent procedure.

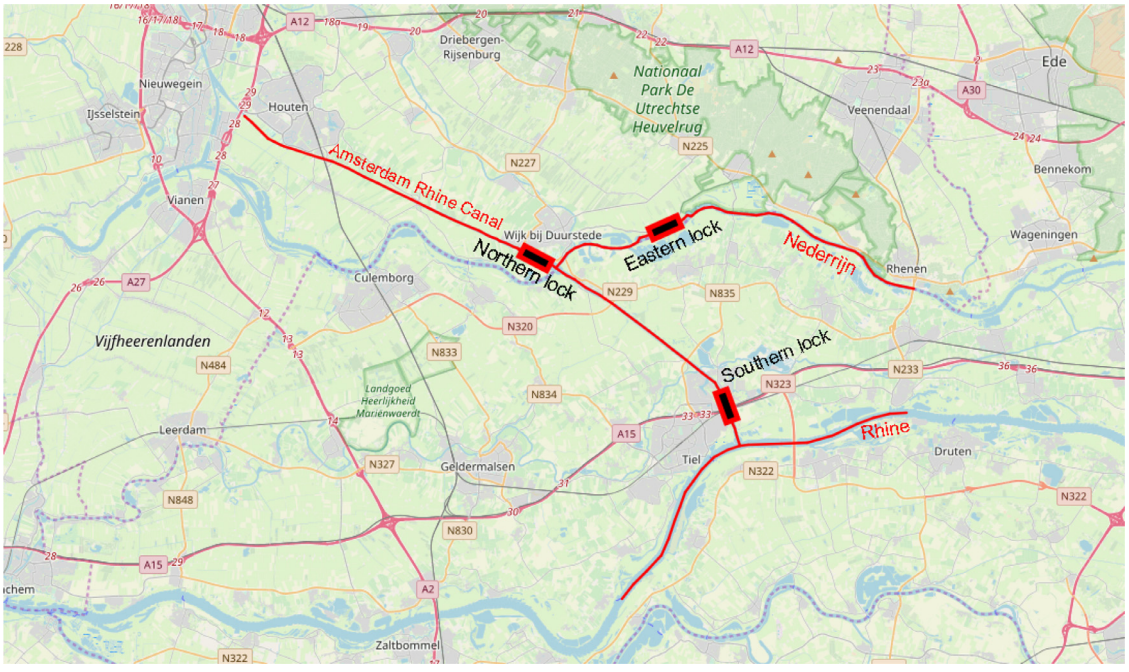

Fig. 4. Map visualization of the Amsterdam Rhine Canal case.

Table 3
Average number of vessels processed by each lock on each day.

Day	Total	East	North	South
Monday	131.92	27.58	110.33	27.58
Tuesday	143.17	30.25	122.42	95.50
Wednesday	142.62	25.85	120.77	97.69
Thursday	150.69	28.69	125.76	98.46
Friday	120.17	20.83	83.33	83.33
Saturday	77.08	8.42	64.25	57.75
Sunday	55.00	4.47	45.83	40.50
Total	**117.92**	**98.98**	**80.15**	**21.04**

Table 4
overview of all lock parameters.

Lock	**Processing Time** (T_l)	**Capacity** (C_l)	**# Chambers** (B_l)
Eastern	22	3	1
North	23	4	2
South	22	3	2

in instances that cover a single day on the river. This means that there are 84 observation days/instances.

The dataset contains vessel characteristics, such as vessel ID and type, and the real-time location of the vessel. These locations are given in the form of GPS coordinates and are recorded every two minutes. In the following section, we give a description of the preprocessing steps, applied to the dataset.

First, a graph was constructed of the specified area with sequences of nodes forming the river segments. Second, GPS positions of individual vessels have been mapped onto the network to track their movements. For each vessel, the first and the last node contained in the dataset are defined as entry and exit points. The timestamps of first and last occurrence are used as arrival time and deadline, respectively. A vessel that stops at a point in the network and continues its journey after some time period (e.g., after lunch break) was replaced by two. Then, we computed the trajectory for each vessel in order to define the sequence of locks (and their directions) that it has to traverse. Lastly, the speed limits were extracted by taking an upper quantile over the measurements of all speed for each type of vessel.

The parameters defining a lock have been attained in the following way. First, for each lock, a set of nodes located at the position of the lock, were specified. Then, the entry and exit times of vessels on these set of nodes were tracked and used to estimate capacity and processing time of lockages. Vessels that were subject to strong noise have been removed from the estimation process. Finally, the median has been taken over all available observation days for each lock individually.

The deadline for each vessel is determined by the physical time the vessel spent in the system. This enables us to perform the heuristic on the data and estimate the potential, real-life improvement of the system. On the other hand, the estimates for a lock's parameter are subject to high variance. There exist situations in which vessels pass the river segment very quickly by facing a low amount of congestion and a low processing time of locks. By setting the lockage duration equal to the median of all representative vessels in the real system, the original vessel movements are considered infeasible by the model. As a result, there exists a subset of vessels in every instance that will not be able to pass the river section given these specifications in the theoretical model. Throughout the numerical experiments, we assume that vessels are homogeneous in size and fuel consumption.

An overview of the average number of vessels processed by each lock on a daily basis is given in Table 3. Furthermore, the estimates of the lock parameters are given in Table 4.

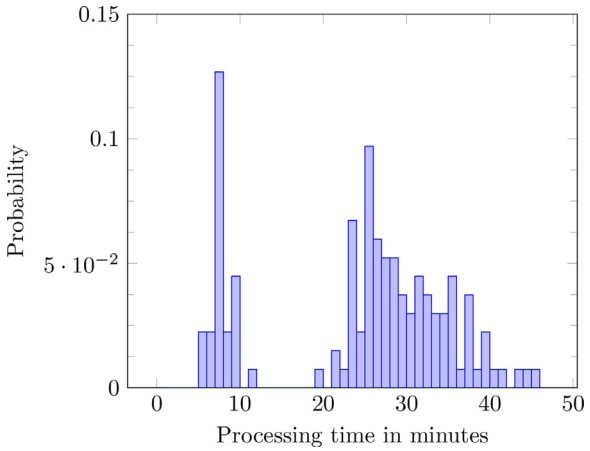

Fig. 5. Realisation of processing time on January, 31, 2019.

5.3. Limitations of the real-time data

Working with real AIS data has its limitations. As these limitations have an impact on how the numerical results presented in this section should be interpreted, we first take the time to discuss them in more detail.

Due to inaccuracy and missing information, it turned out impossible to accurately determine the waiting and processing time of vessels at the locks. The observed waiting time in front of the locks is unreasonably long. This suggests that skippers also use these areas for their required breaks. This makes it hard to distinguish waiting time related to congestion and time taken for breaks. As we currently do not assume that skippers can take a break when solving the speed optimization problem, our algorithm uses the additional available time to reduce the speed on certain river segments. In the results, this will lower total fuel consumption compared to the real-life benchmark, which would give an unfair advantage for our algorithm. Note, however, that these breaks could easily be incorporated in the model formulation if skippers would declare and plan these breaks ahead.

Additionally, we fixed the lockage duration (i.e., the time it takes to process a batch of vessels at the lock) to its median value. The lockage durations for individual vessels in the data, however, deviate substantially from our fixed value. This can clearly be observed in Fig. 5, which provides a graphical representation of the realised processing times for the northern lock on January 31, 2019. Similar assumptions were also made when determining the maximum sailing speed on each river segment. As a result, our mathematical model formulation has a higher minimum time to complete a planned itinerary for some vessels than the original travel time, and therefore larger then the imposed deadline. This means that, by construction, the vessels would be flagged as infeasible to schedule by our solution procedure. These observations would thus give an unfair advantage to the real-life benchmark data.

These limitations imply that policy makers should be careful when relying solely on AIS data to support their decision making process. To allow a fair quality assessment of the proposed heuristic, we therefore solely compare the aggregated fuel consumption of the schedule with the optimal solution generated by the MILP presented above.

5.4. Peformance of the local search heuristic

To evaluate the performance of our local search heuristic, we compare the results to the solutions obtained by the MILP. All numerical experiments have been executed on a Intel(R) Xeon(R)

Table 5
Computational performance of the local search heuristic in comparison to the MILP. (49 Instances).

# vessels	Cost ratio	Computation time (s)		MILP feasible	MILP optimum
		MILP	Local Search	solution found	found (gap)
10	1.046	141.57	2.53	100%	100% (0%)
20	1.015	5008.91	4.68	100%	73.46% (68.01%)
30	0.991	5608.60	7.19	100%	55.1% (77.65%)
40	0.983	6822.69	9.38	85.7%	26.47% (81.85%)
full fleet (avg. 117.92)	–	–	17.62	0%	–

Table 6
Computational performance of the local search heuristic in comparison to instances, where MILP found the optimum.

# vessels	Cost ratio	Computation time (s)	
		MILP	Local Search
10	1.046	141.57	2.73
20	1.048	2111.93	4.32
30	1.053	3221.139	7.14
40	1.074	4919.34	10.14

Gold 6126 CPU 2.60GHz (2 CPUs), 2.6GHz with 6144MB RAM memory.

The MILP was implemented in CPLEX 12.6.3.0. Following the approach of [13], the MILP was further equipped with a set of valid inequalities drastically improving the computation time. We limited the running time of the MILP solver to 7200 s (2 h) per instance. In order to limit the size of the instances, only a fraction of all vessels have been sampled for each day of observation. To ensure the working of the MILP without further modifications, we restricted ourselves to instances that are feasible given the given the parameter settings, as discussed above.

For the heuristic, we apply the initialization phase after which the balancing operator is called at most four time. Extensive numerical experiments on the full fleet of vessels have shown that the total fuel consumption is not decreasing further when increasing the number of iterations of the balancing operator.

The results of the experiments are summarized in Tables 5 and 6. In Table 5, the columns specify the average cost ratio of the heuristic schedule compared to the MILP solution, as well as the running times for both methods. The next column specifies on how many instances the MILP solver found a feasible solution (not necessarily an optimal one). The last column shows on how many instances the solver found an optimal solution and it shows (in brackets) the gap between the lower and the upper bounds. The solver is able to find exact solutions only for instances smaller than 40 vessels, which is far below a realistic instance size. At the same time, the heuristic finds schedules that are of high quality. On the entire fleet, it took on average 17.62 s for the local search heuristic to produce a schedule for all vessels. From these 17.62 s, 7.14 s were required to produce the initial schedule and 10.48 additional seconds were used to apply the four iteration of the speed balancing operator. The balancing procedure was able to decrease the fuel consumption of the initial schedule by 4.05%. Finally, we see that the gap between the bounds varies from 68.01% to 81.85% for 20 and 40 vessels, respectively. Table 6 shows the average cost ratio of the heuristic schedule compared to the MILP schedule and their running times for the instances in which the MILP did find an optimal solution. On the smallest input sizes the heuristic is 4.6% away from the optimal solutions, while on the biggest instance it is 7.4%.

One could observe that the schedule of starting times of lockages is fixed once for all during the initialization phase of our algorithm, which restricts the search space for solutions to the problem. However, as all vessels have a random arrival time when entering the system, a shift in the schedule is likely to level out over time. To verify this assumption, we executed a small scale computational experiment with a constant shift to all lockages, and we haven't seen any significant changes in the results.

5.5. Efficiency versus service level

The limitations of the real-life dataset imply that only a fraction of all vessels can actually trespass the river segment within their indicated deadlines, as discussed in Section 5.3. In this Section, we investigate the trade-off between fuel consumption and service level, defined as an extension of the deadline by α%. The effect of α on the aggregated fuel consumption is plotted in Fig. 6.

For each infeasible vessel (i.e., a vessel that cannot be scheduled to arrive before its deadline because of speed restrictions and fixed lockage durations), we advice a speed such that the vessel's arrival time is as early as possible.

When the value of α increases, the balancing operator becomes less restrictive as service at each lock can be postponed further for each vessel. As a result, the average advised speed will also be lower. The strong decay of the fuel consumption (seen in Fig. 6) implies the increase in sailing time for vessels can be used as a regulatory instrument by policy makers to decrease the total fuel consumption and therefore the total emission considerably.

5.6. Impact of lock efficiency on aggregated fuel consumption

Due to the variation in the processing time of the locks in real-life, the question that arises is how the aggregated fuel consumption is influenced by the lock parameters. The aggregated fuel consumption and the fraction of infeasible vessels are plotted against the difference in lockage duration, denoted by Δ_p, in Fig. 7.

Note that a decreasing lockage duration is correlated with decreasing costs and increasing fraction of feasible vessels. A lower lockage duration implies that the local search has more flexibility, and therefore the gap between the initial schedule and an improved schedule increases. If it is possible to improve the efficiency of locks by advising regulators, it would translate to a direct increase in efficiency and a reduction of aggregated fuel consumption of all vessels.

6. Conclusion and further research

In this work, we introduced a local-search-based heuristic for the lock scheduling problem on a river network and extended the known mathematical programming formulation of [13] to incorporate more realistic features and conditions of the problem. The heuristic algorithm and a straightforward implementation of the

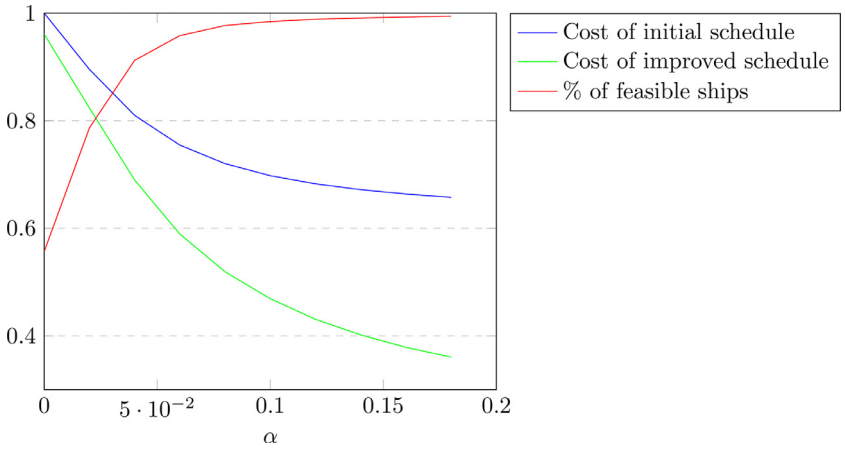

Fig. 6. Impact of relaxing deadline by percentage α on solution quality.

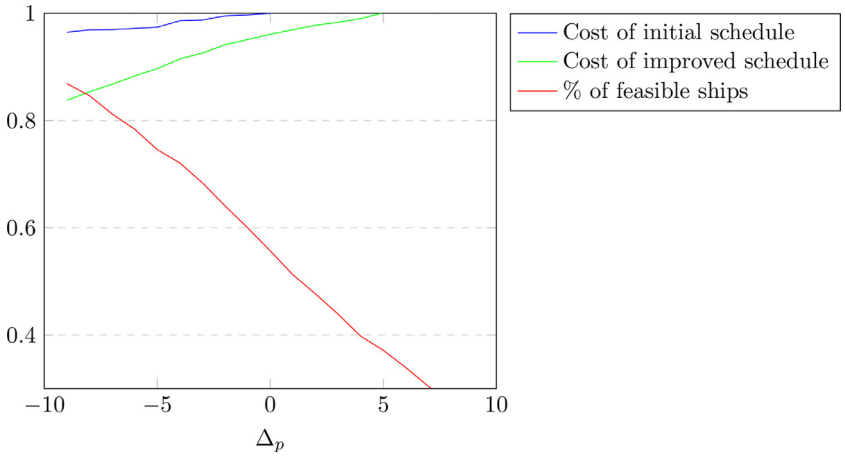

Fig. 7. Impact of change in lockage duration on solution quality.

mathematical formulation were compared using computational experiments on real AIS dataset from a section of the Dutch river network. The heuristic is able to find schedules for large instances of the lock scheduling problem in very reasonable computation time. The mathematical programming formulation is incapable of tackling the large real-life instances. Thus, the suggested heuristic has a potential to become an algorithm for the lock operators and vessel skippers to regulate lock congestions and adjust sailing velocities in order to optimize the fuel consumption and CO2 emission. Furthermore, we have evaluated the effect of varying the service level and the processing time of lockages on the total fuel consumption of the vessels.

In practice, due to unforeseen circumstances, vessels may miss their assigned lockages due to a delay on arrival time into the system or some delay within the system. Therefore, for practitioners it would be interesting to see how such perturbations would damage the quality of a given schedule. The design and analysis of such numerical experiments go beyond the scope of this paper and we propose this as further research to enhance the applicability of our approach.

Furthermore, our results were based on the strong assumption that overtaking is allowed on the considered rivers. In practice, however, this might not always be possible due to narrow river segments or traffic regulations. The integration of such features is both of, theoretical and applied, values and has not been considered so far.

Enhancing the transportation sector by digitalisation and development of appropriate tools would contribute to the sustainability goals imposed by the European Comission. Further refinement and extension of models such as ours may lead to an increase in attractiveness and awareness to practitioners, and thus lead to a transportation sector that is more prepared for today's environmental challenges.

CRediT authorship contribution statement

Julian Arthur Pawel Golak: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing – original draft, Visualization. **Christof Defryn:** Conceptualization, Methodology, Writing – review & editing. **Alexander Grigoriev:** Conceptualization, Methodology, Writing – review & editing, Supervision.

Acknowledgment

First and foremost, we would like to thank Freija van Lent for fruitful discussions on writing and layout. We are also grateful to the cooperating partner "Trapps Wise B.V." for providing us expert insight and the Brightlands Institute for Smart Society for enabling this cooperation. Furthermore, we would like to thank the reviewers for their thorough reading and providing numerous suggestions for improvement of the article.

References

[1] Aghalari A, Nur F, Marufuzzaman M. Solving a stochastic inland waterway port management problem using a parallelized hybrid decomposition algorithm. Omega 2021;102:102316.

[2] Alfandari L, Davidović T, Furini F, Ljubić I, Maraš V, Martin S. Tighter MIP models for barge container ship routing. Omega 2019;82:38–54.
[3] An F, Hu H, Xie C. Service network design in inland waterway liner transportation with empty container repositioning. Eur Transp Res Rev 2015;7(2):9.
[4] Braekers K, Caris A, Janssens GK. Optimal shipping routes and vessel size for intermodal barge transport with empty container repositioning. Comput Ind 2013;64(2):155–64.
[5] Buchem M, Golak JAP, Grigoriev A. Vessel velocity decisions in inland waterway transportation under uncertainty. Eur J Oper Res 2022;296(2):669–78.
[6] Defryn C, Golak JAP, Grigoriev A, Timmermans V. Inland waterway efficiency through skipper collaboration and joint speed optimization. Eur J Oper Res 2021;292(1):276–85.
[7] European Commission. Roadmap to a single European transport area - towards a competitive and resource efficient transport system. https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:52011DC0144&from=EN, Accessed 09 September 2020; 2011.
[8] European Statistical Office. Freight transport statistics - modal split. 2020. Extracted from https://ec.europa.eu/eurostat/statistics-explained/pdfscache/1142.pdf.
[9] Ji B, Zhang D, Yu SS, Fang X. An exact approach to the generalized serial-lock scheduling problem from a flexible job-shop scheduling perspective. Comput Oper Res 2021;127:105164. doi:10.1016/j.cor.2020.105164. https://www.sciencedirect.com/science/article/pii/S0305054820302811
[10] Ji B, Zhang D, Yu SS, Kang C. Mathematical programming models for scheduling multiple cascaded waterway locks. Comput Ind Eng 2021;156:107289. doi:10.1016/j.cie.2021.107289. https://www.sciencedirect.com/science/article/pii/S0360835221001935
[11] Ji B, Zhang D, Yu SS, Zhang B. Optimally solving the generalized serial-lock scheduling problem from a graph-theory-based multi-commodity network perspective. Eur J Oper Res 2021;288(1):47–62. doi:10.1016/j.ejor.2020.05.035. https://www.sciencedirect.com/science/article/pii/S0377221720304811
[12] Nur F, Marufuzzaman M, Puryear SM. Optimizing inland waterway port management decisions considering water level fluctuations. Comput Ind Eng 2020;140:106210.
[13] Passchyn W, Briskorn D, Spieksma FCR. Mathematical programming models for lock scheduling with an emission objective. Eur J Oper Res 2016;248(3):802–14. doi:10.1016/j.ejor.2015.09.012.
[14] Passchyn W, Briskorn D, Spieksma FCR. No-wait scheduling for locks. INFORMS J Comput 2019;31(3):413–28.
[15] Passchyn W, Coene S, Briskorn D, Hurink JL, Spieksma FCR, Berghe GV. The lockmaster's problem. Eur J Oper Res 2016;251(2):432–41. doi:10.1016/j.ejor.2015.12.007.
[16] Petersen ER, Taylor AJ. An optimal scheduling system for the Welland canal. Transp Sci 1988;22(3):173–85.
[17] Prandtstetter M, Ritzinger U, Schmidt P, Ruthmair M. A variable neighborhood search approach for the interdependent lock scheduling problem. In: Ochoa G, Chicano F, editors. Evolutionary computation in combinatorial optimization - 15th European conference, EvoCOP 2015, Copenhagen, Denmark, April 8–10, 2015, Proceedings. Lecture notes in computer science, vol. 9026. Springer; 2015. p. 36–47. doi:10.1007/978-3-319-16468-7_4.
[18] Smith LD, Nauss RM, Mattfeld DC, Li J, Ehmke JF, Reindl M. Scheduling operations at system choke points with sequence-dependent delays and processing times. Transp Res Part E 2011;47(5):669–80. doi:10.1016/j.tre.2011.02.005.
[19] Smith LD, Sweeney DC, Campbell JF. Simulation of alternative approaches to relieving congestion at locks in a river transportion system. J Oper Res Soc 2009;60(4):519–33. doi:10.1057/palgrave.jors.2602587.
[20] Ting C-J, Schonfeld P. Control alternatives at a waterway lock. J Waterway Port Coastal Ocean Eng 2001;127(2):89–96. doi:10.1061/(ASCE)0733-950X(2001)127:2(89).
[21] Verstichel J, De Causmaecker P, Spieksma F, Berghe GV. The generalized lock scheduling problem: an exact approach. Transp Res Part E 2014;65:16–34. doi:10.1016/j.tre.2013.12.010.
[22] Verstichel J, De Causmaecker P, Spieksma FCR, Berghe GV. Exact and heuristic methods for placing ships in locks. Eur J Oper Res 2014;235(2):387–98. doi:10.1016/j.ejor.2013.06.045.
[23] Xia Z, Guo Z, Wang W, Jiang Y. Joint optimization of ship scheduling and speed reduction: a new strategy considering high transport efficiency and low carbon of ships in port. Ocean Eng 2021;233:109224. doi:10.1016/j.oceaneng.2021.109224. https://www.sciencedirect.com/science/article/pii/S0029801821006521
[24] Yang Z, Shi H, Chen K, Bao H. Optimization of container liner network on the yangtze river. Marit Policy Manage 2014;41(1):79–96.
[25] Zhen L, Wang K, Wang S, Qu X. Tug scheduling for hinterland barge transport: abranch-and-price approach. Eur J Oper Res 2018;265(1):119–32.