Mathematical programming models for scheduling locks in sequence

Ward Passchyn[1], Dirk Briskorn[2], and Frits C.R. Spieksma[1]

1 **KU Leuven**
2 **Bergische Universität Wuppertal**

Abstract

We investigate the scheduling of series of consecutive locks. This setting occurs naturally along canals and waterways. We describe a problem that generalizes different models that have been studied in literature. Our contribution is to (i) provide two distinct mathematical programming formulations, and compare them empirically, (ii) show how these models allow for minimizing emission by having the speed of a ship as a decision variable, (iii) to compare, on realistic instances, the optimum solution found by solving the models with the outcome of a decentralized heuristic.

1998 ACM Subject Classification G.2.1 [Discrete Mathematics]: Combinatorics, G.2.3 [Discrete Mathematics]: Applications

Keywords and phrases Mixed Integer Programming, Inland Waterways, Lock Scheduling

Digital Object Identifier 10.4230/OASIcs.ATMOS.2014.92

1 Introduction

On many inland waterways, locks are required to ensure a suitable water level for navigation. Typically, and notably when the waterway traffic density is high, locks act as bottlenecks, introducing waiting time for ships that pass through these canals and waterways. We consider here the setting where a series of consecutive locks is arranged in a sequence along a canal. In this problem setting, ships travel in both directions and each lock acts as a single server which handles both the upstream and the downstream traffic. Lock operations must alternate between upwards (downstream to upstream) and downwards (upstream to downstream) movements. Results on scheduling a single lock in order to minimize ship waiting times exist in the literature. One goal of this work is to investigate the performance gain that results from centralizing the decision-making for the sequence of locks, and coordinating the lock movements so that ships move more fluently through the system. Indeed, if each of the locks schedules its movements separately, some flexibility may be lost in obtaining a globally optimal solution e.g. when waiting time at one of the locks cannot be avoided, it may be beneficial for a ship to incur this waiting time as early as possible in order to improve the performance of the subsequent locks. In a centralized approach, we can avoid this problem by incorporating the decisions of each individual lock in obtaining a global schedule that minimizes the total waiting time over the entire length of the canal.

Section 2 provides a brief overview of related literature and known results. We also cover some literature as context for the problem, e.g. fuel consumption and greenhouse gas emissions, since our models have the speed of a ship as a decision variable. A formal definition of the problem is given in section 3. We aim to provide a general problem description which may be easily extended by incorporating case-specific constraints or alternative objective functions. For this general problem setting, we propose two integer programming models (section 4). Next, we outline a heuristic which uses the solution procedure for a single lock as a building block. The heuristic combines the decentralized decision-making procedures for the individual locks along the canal into a global schedule that serves each of the ships. Finally, section 6 covers some computational experiments to investigate the performance of the different exact methods, as well as the difference in solution quality when compared to the decentralized heuristic.

2 Related literature

Applying optimization techniques in the context of scheduling locks is not new in literature. An early example is the case of the Welland Canal in North America, which allows ships to bypass the Niagara Falls. The St. Lawrence Seaway Authority, which maintains the canal, faced increasing congestion at the locks along the canal. In [6], an integer programming model for this situation is described. The authors also discuss a dynamic programming model for scheduling the operations of a single lock, and extend this model to a heuristic that yields an operating schedule for the series of locks along the Welland Canal.

Due to the computational effort involved, a majority of works in the literature restricts the attention to simulation models and heuristic solutions. In [8], for example, simulation models are used in order to aid policy decisions to reduce congestion on the Upper Mississippi River. Different ship sequencing policies for the Mississippi river are also evaluated in [10]. A notable difference between the Mississippi River and the problem setting considered here is that barges on the Mississippi River are typically not handled in batches and may even require more than one lockage due to their length exceeding the lock capacity.

A different approach is proposed in [2], where the waiting time at a single lock is minimized while allowing a batch of multiple ships to be grouped together and processed in a single lockage. The authors cover a number of problem extensions to the initial problem setting and perform some computational tests to investigate the performance of heuristics. In [11], an integer programming model is presented that minimizes the waiting time for ships at a single lock, which may consist of parallel chambers, including the aspect of placing the ships inside the chambers. A model for traffic optimization, including the sequencing of ships and scheduling locks, has also been proposed for the Kiel canal, connecting the Baltic Sea to the North Sea [3]. An implementation for a lock scheduling heuristic, including ship placement and parallel chambers, is available online [5]. The importance of an efficient lock operating strategy is also noted in [1], where lock scheduling decisions strongly affect the simulated waiting time and efficient decision rules for lock operating are suggested as future work.

Results on obtaining optimum solutions to a system of multiple locks as a whole, are more scarce in the literature. The potential of a centralized approach to scheduling has, however, recently attracted more attention in the field. The Dutch waterway management organization Rijkswaterstaat, for example, is shifting its focus from decentralized lock operations towards the fluent operation of certain ‘corridors’ as a whole [4].

A different objective could be to minimize the fuel consumption for ships passing through the waterway system. While the fuel consumption may be an important economical factor for ship operators, the related emission of greenhouse gases may also be a an optimization criterion for governments or waterway organizations. In recent years, the operational speed of intercontinental container ships has decreased to improve fuel efficiency, a practice referred to as ‘slow steaming’. On inland waterways however, ships are likely to incur waiting time near bottlenecks such as locks. This provides ships with the opportunity of decreasing their maximal speed on each of the sections along the canal while their total time spent inside the

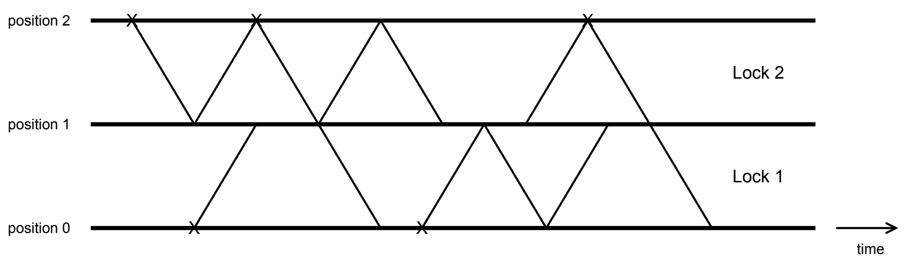

Figure 1 Illustration of a problem instance with $L = 2$, $S = 5$, and a feasible solution. Time passes from left to right.

canal remains the same. A model for the time-varying vehicle routing problem is proposed in [7], where the goal is to minimize the greenhouse gas emissions, which can be directly related to the vehicle speed. On inland waterways, the strategy of reducing ship speed to avoid idle time was considered in [9] and is reported to yield significant economic benefits.

3 Problem definition

We give here a formal statement of the problem we are considering. For clarity of presentation, all known parameters are represented in uppercase, decision variables in lowercase, and sets in calligraphic script. Given is a series of L consecutive locks, for example, along a canal. We refer to them using the set $\mathcal{L} = \{1, \ldots, L\}$. Over time, S ships arrive at either end of the canal. The set $\mathcal{S} = \{1, \ldots, S\}$ allows us to uniquely identify these ships. Each of the ships travels to the end of the canal opposite to its arrival. As a consequence, a ship either arrives on the downstream side of lock 1 and must pass each of the locks $1, \ldots, L$ in order, or a ship arrives at lock L and must pass all locks in the order $L, \ldots, 1$. Each lock $\ell \in \mathcal{L}$ has a strictly positive lockage time P_ℓ, which is the time needed for a lock to change its position and to allow ships to leave and enter as needed. We assume that the lockage time is independent of the number of ships in the lock and their direction of travel. Locks also have a positive capacity C_ℓ which gives, for all $\ell \in \mathcal{L}$, an upper bound on the number of ships that can simultaneously be present in lock ℓ. We do not assume an initial position for the locks, i.e. each lock may start at either its upstream or downstream position. The locks are separated by a section of canal with length S_ℓ with $\ell \in \mathcal{L} \setminus \{L\}$, where S_ℓ equals the distance between locks ℓ and $\ell + 1$. Each ship $s \in \mathcal{S}$ may travel at an arbitrary speed contained in the interval $[V_s^{\min}, V_s^{\max}]$. While the ships can travel at a different speed on different sections of the canal, we assume that each ship maintains a constant speed within each section. Each ship $s \in \mathcal{S}$ may also have an imposed deadline D_s, before which it must have left the last lock that it needs to pass. In what follows, we assume that all arrival times and positions are known.

We can graphically represent an instance to this problem as depicted in Figure 1. The tilted lines correspond to a set of possible movements for each of the locks. A feasible solution is a schedule specifying the time when each of the ships is moved by each of the locks so that all ships arrive at their destination. Any feasible solution can thus also be easily visualized. It must hold that, for any lock, the lock's movements alternate between the upwards and downwards direction, and none of the lock's movements overlap. In the schedule depicted in Figure 1, once a ship is in a given position, it may enter the next lockage in its direction of travel.

Table 1 Summary of notation.

$\mathcal{L}$	$= \{1,\ldots, \mathrm{L}\}$	the set of all locks,
$\mathcal{S}$	$= \{1,\ldots, \mathrm{S}\}$	the set of all arriving ships,
$\mathcal{U}$		the set of all ships arriving on the upstream side,
$\mathcal{D}$		the set of all ships arriving on the downstream side,
P_ℓ	$(\ell \in \mathcal{L})$	the processing time, i.e. lockage time, for lock ℓ,
C_ℓ	$(\ell \in \mathcal{L})$	the lock capacity for lock ℓ,
S_ℓ	$(\ell \in \mathcal{L} \setminus \{L\})$	the length of the canal section separating lock ℓ and $\ell + 1$,
A_s	$(s \in S)$	the arrival time of ship s,
$V_s^{\min}$	$(s \in \mathcal{S})$	the minimum speed attainable by ship s,
$V_s^{\max}$	$(s \in \mathcal{S})$	the maximum speed attainable by ship s,
D_s	$(s \in \mathcal{S})$	the deadline for ship s.

The goal is to find a solution which performs best with respect to some predetermined objective function. A relevant measure is to minimize the total flow time $\sum_{s\in\mathcal{S}} F_s = \sum_{s\in\mathcal{S}} c_s - A_s$, where c_s equals the completion time of ship s.

A different objective function concerns the emission of greenhouse gases, which is closely related to the fuel consumption, and depends on the ship's speed. We assume that we have a known emission function $E(v)$ which expresses the emission of pollutants, in tons per km, as a function of the ship speed v for $v > 0$. Note that this function E depends on many factors and is generally non-linear. Let $v_{s,p}$ denote the speed of ship s along segment p. The total emission of greenhouse gases E_{tot} can then be computed as follows:

$$E_{\text{tot}} = \sum_{s\in\mathcal{S}} \sum_{\ell\in\mathcal{L}\setminus\{L\}} S_\ell E(v_{s,\ell}).$$

A similar approach can be applied in order to minimize fuel consumption, which may be more desirable from the shipper's point of view. The problem parameters are summarized in table 1.

4 Exact solutions

We introduce here two distinct integer programming models that find an optimal solution to the general problem described above. While both models solve the same problem, their different formulations have advantages as well as disadvantages with regards to computation time. For both models, we also introduce valid inequalities that tighten the LP relaxation and may thus speed up the process of finding an optimal solution. For a comparison of the difference in performance between the formulations and their extensions, we refer to section 6.1.

To state the travel time, which appears in the constraints of both models, as a linear expression of the variables, we introduce the variables $\bar{v}_{s,p}$, which equal the reciprocal of $v_{s,p}$. The travel time for ship s along section p then equals $\bar{v}_{s,p}S_p$. To characterize the emissions, we can compute the function $\bar{E}(\bar{v})$, which expresses the emissions as a function of the reciprocal of ship speed.

Note that even with S_p and $v_{s,p}$ integral, the travel time for some sections may be fractional. From a practical point of view, however, it might not make sense to schedule lock movements with a higher precision than the unit in which the arrival times are expressed, e.g. minutes. The time-indexed model described below allows lockages to start only at integral moments in time, whereas the second model allows arbitrary starting times for the lock movements.

4.1 MIP 1: Time-indexed formulation

The first model introduces a variable for each moment in time when a lockage may start. For this, we introduce the set $\mathcal{T} = \{0, \ldots, T\}$ with all integral moments in time where a lockage may start. We have T as an upper bound on the latest starting time.

In addition to the variables $\bar{v}_{s,\ell}$ introduced above, we define the following binary decision variables: For each $s \in \mathcal{S}$, $\ell \in \mathcal{L}$, $t \in \mathcal{T}$, let

$$x_{s,\ell,t} = \begin{cases} 1 & \text{if, at time } t\text{, lock } \ell \text{ starts a lockage with ship } s \text{ inside the lock,} \\ 0 & \text{otherwise.} \end{cases}$$

We state below the time-indexed mixed integer programming model as a whole, before discussing the different equations separately:

$$\text{Minimize} \sum_{s\in\mathcal{S}} \Big(c_s - A_s\Big) \quad \text{or} \quad \text{Minimize} \sum_{s\in\mathcal{S}} \sum_{\ell\in\mathcal{L}\setminus\{L\}} S_\ell \bar{E}(\bar{v}_{s,\ell}) \tag{1}$$

Subject to:

$$\sum_{t=A_s}^{D_s-P_1} x_{s,\ell,t} = 1 \qquad \forall s \in \mathcal{U}, \ell \in L \tag{2}$$

$$\sum_{t=A_s}^{D_s-P_L} x_{s,\ell,t} = 1 \qquad \forall s \in \mathcal{D}, \ell \in L \tag{3}$$

$$c_s = \sum_{t\in\mathcal{T}} \big(t\, x_{s,1,t} + P_1\big) \qquad \forall s \in \mathcal{U} \tag{4}$$

$$c_s = \sum_{t\in\mathcal{T}} \big(t\, x_{s,L,t} + P_L\big) \qquad \forall s \in \mathcal{D} \tag{5}$$

$$\sum_{t\in\mathcal{T}} (t\, x_{s,\ell,t}) - \sum_{t\in\mathcal{T}} (t\, x_{s,\ell+1,t}) \geq P_{\ell+1} + \bar{v}_{s,\ell} S_\ell \qquad \forall s \in U, \forall \ell \in \mathcal{L} \setminus \{L\} \tag{6}$$

$$\sum_{t\in\mathcal{T}} (t\, x_{s,\ell,t}) - \sum_{t\in\mathcal{T}} (t\, x_{s,\ell-1,t}) \geq P_{\ell-1} + \bar{v}_{s,\ell-1} S_{\ell-1} \qquad \forall s \in \mathcal{D}, \forall \ell \in \mathcal{L} \setminus \{1\} \tag{7}$$

$$x_{s_1,\ell,t} + \sum_{\tau=t-P_\ell+1}^{t+P_\ell-1} x_{s_2,\ell,\tau} \leq 1 \qquad \forall \ell \in \mathcal{L}, s_1 \in \mathcal{U}, s_2 \in \mathcal{D}, t \in \mathcal{T} \tag{8}$$

$$x_{s_1,\ell,t} + \sum_{\tau=t-2P_\ell+1}^{t-1} x_{s_2,\ell,\tau} + \sum_{\tau=t+1}^{t+2P_\ell-1} x_{s_2,\ell,\tau} \leq 1 \qquad \forall \ell \in \mathcal{L}, s_1 \in \mathcal{U}, s_2 \in \mathcal{U}, t \in \mathcal{T} \tag{9}$$

$$x_{s_1,\ell,t} + \sum_{\tau=t-2P_\ell+1}^{t-1} x_{s_2,\ell,\tau} + \sum_{\tau=t+1}^{t+2P_\ell-1} x_{s_2,\ell,\tau} \leq 1 \qquad \forall \ell \in \mathcal{L}, s_1 \in \mathcal{D}, s_2 \in \mathcal{D}, t \in \mathcal{T} \tag{10}$$

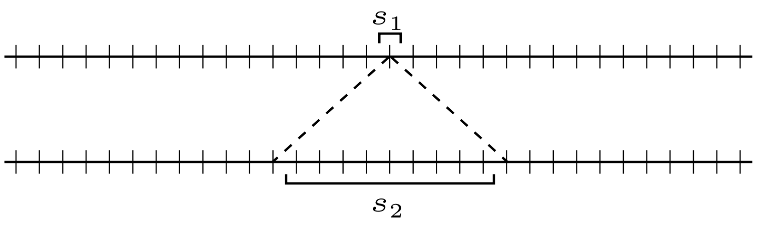

Figure 2 For any s_1 and s_2 travelling in opposite directions, ship s_2 cannot start a movement in the indicated interval if the x variable for s_1 equals one.

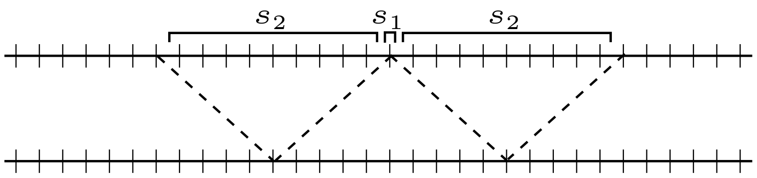

Figure 3 For any s_1 and s_2 travelling in the same direction, ship s_2 cannot start a movement in the indicated intervals if the x variable for s_1 equals one.

$$\sum_{s \in S} x_{s,\ell,t} \leq C_\ell \qquad \forall \ell \in \mathcal{L}, t \in \mathcal{T} \tag{11}$$

$$\frac{1}{V_s^{\max}} \leq \bar{v}_{s,\ell} \leq \frac{1}{V_s^{\min}} \qquad \forall s \in \mathcal{S}, \ell \in \mathcal{L} \setminus \{L\} \tag{12}$$

$$x_{s,\ell,t} \in \{0,1\} \qquad \forall s \in \mathcal{S}, \ell \in \mathcal{L}, t \in \mathcal{T} \tag{13}$$

For a schedule to be feasible, each ship should pass each of the locks and on time to meet its deadline, as imposed by inequalities (2) and (3). Further, all locks must be passed in the correct order, i.e. a ship must arrive at a lock before it can enter that lock. We achieve this by imposing constraints (6)-(7).

Additionally, a lockage can only start when the lock is in the appropriate position and not currently moving, i.e. lockages of the same lock should not overlap in time. We impose this by adding the constraints (8)-(10). Figures 2 and 3 give a visual representation of the overlap constraints. Note that constraints (9) and (10), which concern ships on the same side of a lock, allow multiple ships to be handled at the same time.

Finally, the lock capacity and domain restrictions are straightforward to impose by constraints (11)-(12).

We point out that this model contains $O(SLT)$ binary variables, $O(SL)$ real variables, and $O(S^2LT)$ constraints. We refer to Appendix A.1 for an overview of valid inequalities to improve the time-indexed model (1)-(13).

4.2 MIP 2: Lockage-based formulation

A notable disadvantage to the time-indexed model is that the number of (binary) variables grows as the time horizon increases. For a small discretization step or when arrival times are large, the value for the upper bound T and thus the number of variables, and the computation time to find an optimal solution, may grow prohibitively large. We introduce an alternative formulation that does not use a time index for the variables, and instead numbers the possible lockages. It is clear that for each lock, the number of lockages in an optimal solution need not be greater than $2S$. At most we may have one lockage for each of the ships, followed by an empty lockage to switch back to the appropriate position for the next ship. We define the set $\mathcal{M} = \{1, \dots, 2S\}$ to identify the different lock movements. Each lock movement must also be assigned a specific starting time. Note that the number of variables does not increase with T.

In addition to $\bar{v}_{s,p}$ and c_s, we introduce the following decision variables:

$$z_{s,\ell,m} \quad = \begin{cases} 1 & \text{if ship } s \text{ is handled by the } m\text{'th lock movement of lock } \ell, \\ 0 & \text{otherwise.} \end{cases}$$

$t_{\ell,m}$ equals the starting time of the m'th lockage of lock ℓ.

The model is as follows:

$$\text{Minimize} \sum_{s \in \mathcal{S}} \Big(c_s - A_s \Big) \quad \text{or} \quad \text{Minimize} \sum_{s \in \mathcal{S}} \sum_{\ell \in \mathcal{L} \setminus \{L\}} S_\ell \bar{E}(\bar{v}_{s,\ell}) \tag{14}$$

Subject to:

$$D_s \geq c_s \geq t_{1,m} + P_1 - T(1 - z_{s,1,m}) \qquad \forall s \in \mathcal{U}, m \in \mathcal{M} \tag{15}$$
$$D_s \geq c_s \geq t_{L,m} + P_L - T(1 - z_{s,L,m}) \qquad \forall s \in \mathcal{D}, m \in \mathcal{M} \tag{16}$$
$$\sum_{m \in \mathcal{M}} z_{s,\ell,m} = 1 \qquad \forall s \in \mathcal{S}, \ell \in \mathcal{L} \tag{17}$$
$$t_{\ell,m} \geq t_{\ell,m-1} + P_\ell \qquad \forall \ell \in \mathcal{L}, m \in \mathcal{M} \setminus \{1\} \tag{18}$$

$$z_{s_1,\ell,m} + z_{s_2,\ell,m} \leq 1 \qquad \forall s_1 \in \mathcal{U}, s_2 \in \mathcal{D}, \ell \in \mathcal{L}, m \in \mathcal{M} \tag{19}$$
$$z_{s_1,\ell,m-1} + z_{s_2,\ell,m} \leq 1 \qquad \forall s_1, s_2 \in \mathcal{U}, \ell \in \mathcal{L}, m \in \mathcal{M} \setminus \{1\} \tag{20}$$
$$z_{s_1,\ell,m-1} + z_{s_2,\ell,m} \leq 1 \qquad \forall s_1, s_2 \in \mathcal{D}, \ell \in \mathcal{L}, m \in \mathcal{M} \setminus \{1\} \tag{21}$$

$$t_{L,m} \geq z_{s,L,m} A_s \qquad \forall s \in \mathcal{U}, m \in \mathcal{M} \tag{22}$$
$$t_{1,m} \geq z_{s,1,m} A_s \qquad \forall s \in \mathcal{D}, m \in \mathcal{M} \tag{23}$$
$$t_{\ell,m_1} \geq t_{\ell+1,m_2} + P_{\ell+1} + \bar{v}_{s,\ell} S_\ell - T(2 - z_{s,\ell,m_1} - z_{s,\ell+1,m_2}) \qquad \forall s \in \mathcal{U}, \ell \in \mathcal{L} \setminus \{L\}, m_1, m_2 \in \mathcal{M} \tag{24}$$
$$t_{\ell,m_1} \geq t_{\ell-1,m_2} + P_{\ell-1} + \bar{v}_{s,\ell-1} S_{\ell-1} - T(2 - z_{s,\ell,m_1} - z_{s,\ell-1,m_2}) \qquad \forall s \in \mathcal{D}, \ell \in \mathcal{L} \setminus \{1\}, m_1, m_2 \in \mathcal{M} \tag{25}$$

$$\sum_{s \in S} z_{s,\ell,m} \leq C_\ell \qquad \forall \ell \in \mathcal{L}, m \in \mathcal{M} \tag{26}$$
$$\frac{1}{V_s^{\max}} \leq \bar{v}_{s,\ell} \leq \frac{1}{V_s^{\min}} \qquad \forall s \in \mathcal{S}, \ell \in \mathcal{L} \setminus \{L\} \tag{27}$$
$$z_{s,\ell,m} \in \{0,1\} \qquad \forall s \in \mathcal{S}, \ell \in \mathcal{L}, m \in \mathcal{M} \tag{28}$$
$$t_{\ell,m} \in \mathbb{R}_+ \qquad \forall \ell \in \mathcal{L}, m \in \mathcal{M} \tag{29}$$
$$c_s \in \mathbb{R}_+ \qquad \forall s \in \mathcal{S} \tag{30}$$

Inequalities (15) and (16) ensure that the completion time of each ship, used in the objective function, is consistent with the timing of the last lockage the ship passes through and enforce the deadlines for each ship. Constraint (17) guarantees that each ship passes through each of the locks. Constraint (18) imposes an order on the lockages for each of the locks, and ensures that they do not overlap in time.

The next set of inequalities, (19)-(21), ensures that a lockage can only handle ships in a single direction with each lockage, and that no two consecutive lockages carry ships in the same direction. Note that because the only way to characterize the direction of a lockage is to consider the direction of ships inside the lockage, the direction of lockages need not necessarily alternate when empty lockages are present. Using these constraints, however, the model does guarantee that all non-empty lockages satisfy all requirements for a feasible solution.

Obviously, the locks should be passed in the correct order. This is imposed by specifying that the waiting time each ship incurs at each position must be non-negative. Constraints (22) and (23) ensure this for the outer positions where the arrival times are known, whereas constraints (24) and (25) do the same for the middle positions.

Again, the capacity constraint (26) and the domain restrictions for the variables are straightforward to specify.

The lockage-based model involves $O(S^2L)$ binary variables, $O(SL)$ real variables, and $O(S^3L)$ constraints. We refer to Appendix A.2 for an overview of several valid inequalities for the lockage-based model (14)-(30).

5 Decentralized heuristic

In order to estimate the potential improvement in performance of a centralized schedule over decentralized schedules, we describe here a heuristic procedure that computes a solution schedule based on decentralized decision-making by operators of the individual locks. In what follows, we restrict the problem by assuming that all ships are identical and travel at a known speed. Our goal is to minimize the total flow time. In [2], an efficient procedure is introduced to determine the optimal schedule for a single lock. We will use this procedure on each of the locks separately to obtain a schedule for the sequential system. We are however facing the problem that, with the locks arranged in sequence, the arrival times of ships at a lock are determined by the schedule of any locks that the ships must pass first. Thus, not all arrival times at each of the locks are known initially.

We resolve this problem by iteratively scheduling each of the single locks for those arrival times that are known. We then update arrival information for the following iterations and repeat the process until the solution has converged. A solution has converged when the arrival times computed in the current iteration are equal to the arrival times of the previous iteration, for each lock. An outline of this procedure in pseudo-code is given in table 1, where SLS(U, D) represents the single lock solver procedure which returns a solution to the single lock problem given a set of upstream arrival U and downstream arrivals D, and f_u (f_d) is a function that computes the upstream (downstream) arrival times at a lock given from a given schedule at the previous lock. The input for this procedure consists of the reciprocal of the common ship speed, section lengths, and sets with all arrival times of ships travelling upstream, as well as downstream, at the lock where they enter the system. These sets of arrival times will be denoted by U_1^L and D_1^1 respectively.

Consider for a moment that we have only two locks. We start by scheduling the first lock while only considering those ships for which the arrival time is known. Next, we schedule the second lock, including only the initially available arrival information. After obtaining a schedule for the second lock, we update the arrival times at each of the locks based on the individual schedules for both locks, adjusted with the appropriate lockage duration and

Algorithm 1 Pseudo-code algorithm for the decentralized heuristic

```
Input: $U_1^L, D_1^L, S_\ell \ (\ell \in \mathcal{L} \setminus \{L\}), \bar{v}$
i=1
repeat
  $U_{i+1}^L = U_i^L$
  $D_{i+1}^1 = D_i^1$
  for $\ell \in \mathcal{L} \setminus \{1\}$
    $U_{i+1}^{\ell-1}$ = $f_u$(SLS$(U_i^\ell, D_i^\ell)) + \bar{v}S_{\ell-1}$
  end for
  for $\ell \in \mathcal{L} \setminus \{L\}$
    $D_{i+1}^{\ell+1}$ = $f_d$(SLS$(U_i^\ell, D_i^\ell)) + \bar{v}S_\ell$
  end for
  i = i + 1
until $U_i^l = U_{i-1}^\ell$ AND $D_i^\ell = D_{i-1}^\ell$ for all $\ell \in \mathcal{L}$, OR i = 10L
```

travel times. We now recalculate the individual schedule for each of the locks and keep repeating this procedure until the solution has reached convergence.

Note that this procedure does not correspond entirely to the decisions made in practice. Because the arrival times are obtained from previous iterations, each lock operator thus implicitly takes future decisions of neighbouring locks into account. Since this information is not available in practice, we may expect the iterative procedure to perform better than a typical decentralized solution in practice.

We should also note that there is no guarantee that the procedure converges for every input. In fact, it is possible to construct an instance so that the algorithm cycles infinitely between two solutions. For this reason we add an upper bound on the number of iterations to ensure that the algorithm terminates.

6 Computational study

In this section, we perform some computational tests to analyze the performance of centralized schedules, as well as looking into the performance of the different solution procedures. We first investigate how the valid inequalities described in the appendix impact the LP-relaxation and the solution time. We then apply both the centralized and decentralized procedures to a set of problem instances representative for a real-world setting and estimate the potential gain in performance.

6.1 Comparison of the MIP-models

We implement the different model formulations with the flow time objective in CPLEX (version 12.5.1) and investigate their performance. We generate instances with 3 locks in sequence, each with a capacity of 3. We perform these experiments once for 50 randomly generated instances with a time horizon of 48 units, an expected time between arrivals of 4, and a lockage time of 3. For simplicity, the distance between the locks is considered to be zero. We then multiply all arrival times, the time horizon, and the lockage time for these instances by 10 and repeat the experiments. The first set of instances can thus be considered to have a large unit of time, while the second experiment models the same instances with a time unit that is 10 times smaller. We report the average performance for each of the models in Tables 2 and 3. The time-indexed and lockage-based models are denoted with TI and

Table 2 Averaged model performance for the different MIP models, large time units.

instances solved to optimality within 600s			
TI	TI+	LB	LB+
50	50	4	13
average computation time [s] (unsolved=600s)			
TI	TI+	LB	LB+
12.8	11.8	558.2	466.9
average relative optimality gap			
TI	TI+	LB	LB+
0 %	0 %	1189 %	641 %
average best int to optimum ratio			
TI	TI+	LB	LB+
100 %	100 %	121 %	150 %

LB respectively. A '+' denotes that the valid inequalities discussed in the appendix have been applied. In addition to the valid inequalities, we reduce the search space for each of the 4 models by adding constraints that enforce that ships travelling in the same direction are handled on a first-come first-served basis. We limit the computation time to 10 minutes per instance. The reported optimality gap is expressed relative to the best known optimal solution. Because the optimum value for an instance in the second experiment is necessarily equal to 10 times the optimum value of the corresponding instance in the first experiment, we can also express the ratio of the best found integral solution to the known optimum solution.

For the instances with small time units, neither the TI nor the TI+ model found solutions. Due to the large number of variables and constraints as a result of the large time horizon, even the pre-processing for the time-indexed models was impossible. A large difference in performance thus immediately shows for both model types, depending on the number of variables. As may be expected, the TI and TI+ models perform poorly when the unit of time is small and, consequently, the number of time variables is relatively large. When this is the case, the LB+ model obtains an optimal solution much faster. However, when the number of variables is limited, the TI and TI+ models significantly outperform the lockage-based models. An explanation for this can be found in constraints (15)-(16) and (24)-(25), which are of the big-M type. As a consequence, the lower bound obtained from the LP-relaxation is low, which significantly slows down exact procedures through branch-and-bound or branch-and-cut, such as employed by CPLEX. This is reflected in the large optimality gaps for the lockage-based models. While the lockage-based models allow the modelling of instances with large time horizons, proving optimality becomes difficult. Increasing the time limit per instance to 30 minutes indicated only marginal improvement to the found integral solutions, while only a few additional instances could be proved optimal. It can also be noted that while the LB+ model can prove optimality faster than the LB model, the latter seems to provide better integral solutions for the instances that cannot be solved within the time limit.

Table 3 Averaged model performance for the different MIP models, small time units.

instances solved to optimality within 600s			
TI	TI+	LB	LB+
0	0	4	13
average computation time [s] (unsolved=600s)			
TI	TI+	LB	LB+
600.0	600.0	560.3	467.0
average relative optimality gap			
TI	TI+	LB	LB+
-	-	1137 %	602 %
average best int to optimum ratio			
TI	TI+	LB	LB+
-	-	126 %	172 %

6.2 Comparing centralized and decentralized schedules

In order to estimate the potential benefit of centralizing decision-making, we simulate a centralized as well as a decentralized approach for instances reflecting a realistic problem setting. We generate instances resembling a real-world scenario based on the layout of the Bocholt-Herentals canal in Belgium. The canal is in CEMT class II, and allows ships up to 600 tonnes. It connects to the economically important Albert Canal, and thus to the ports of Antwerp and Liège. Specifically, we focus on three locks between Mol and Dessel, over a total length of 6.2km. The primary reason for selecting this canal segment is that each of the locks consists of a single chamber. Our problem, as defined in section 3, thus closely resembles the setting on this canal segment. Arrival times are generated randomly, with times between arrivals drawn from a geometric distribution to ensure a memoryless arrival process where the arrival times for ships are independent. Our time horizon is 480 minutes, with an expected time between ship arrivals of 30 minutes. Ships arrive on either end of the canal with equal probability. Each lock is assumed to have a capacity of 3, and a lockage time of 30 minutes. We minimize the sum of flow times over all ships. To reduce the search space, we again enforce that all ships travelling in the same direction must be handled on a first-come first-served basis. The solution time for each instance is limited to 30 minutes. The averaged results over 10 instances are presented in Table 4. On using the LB+ model, we observe that a loose upper bound on the number of lockages results in excessively long computation times and low-quality solutions. We impose a limit of S on the number of lockages to improve the values of the found integral solutions. To distinguish this procedure from the LB+ model, we represent it by LB*. The heuristic that repeatedly iterates over single lock instances, as described in Section 5, is denoted by RISL. Table 4 provides an overview of the results, from which an average performance gain of 62.7% can be estimated. This gain comes at the cost of a significant increase in the required computational effort.

Table 4 Results for the LB+ model.

Total waiting time [min]		
LB*	RISL	relative gain
698	1465	52 %
359	1301	72 %
498	1062	53 %
507	1235	59 %
380	1046	64 %
446	1035	57 %
256	918	72 %
87	678	87 %
1377	2384	42 %
529	1644	68 %
Average computation time [s]		
LB*	RISL	
1630.0	0.3	

7 Conclusion and future work

In this work, we formulated the general problem of scheduling a system of locks arranged in a sequence. We described two mixed integer programming models which solve the problem to optimality. Both models can be easily extended to include additional constraints or alternative objective functions such as minimizing the total greenhouse gas emissions, as opposed to the total flow time. An example of a further generalization of the problem would be to attribute different priorities to the ships, and to add weights to the objective function accordingly.

Computational testing confirms that the TI model performs well when the unit of time is chosen sufficiently large. When the unit of time is small, the required computation time for the TI model quickly increases, whereas the LB model suffers significantly less from this drawback. The LB model, however, performs significantly worse than the TI model for the instances with a limited number of time variables.

A different extension with practical relevance that was not discussed here is the setting where ships may enter or leave the system in between the locks. With some additional notation and minor modifications, this may be easily added to the models. Further investigation into variants of this problem, such as those including greenhouse gas emissions, is ongoing.

References

1 A. Caris, G. Janssens, and C. Macharis. A simulation approach to the analysis of intermodal freight transport networks. In *ESM'2007 Proceedings*. EUROSIS, 2007.

2 S. Coene, W. Passchyn, F.C.R. Spieksma, G. Vanden Berghe, D. Briskorn, and J.L. Hurink. The lockmaster's problem. *under review*, 2013.

3 E. Günther, M. E. Lübbecke, and R. H. Möhring. Ship Traffic Optimization for the Kiel Canal. In *TRISTAN VII Book of Extended Abstracts*, 2010.

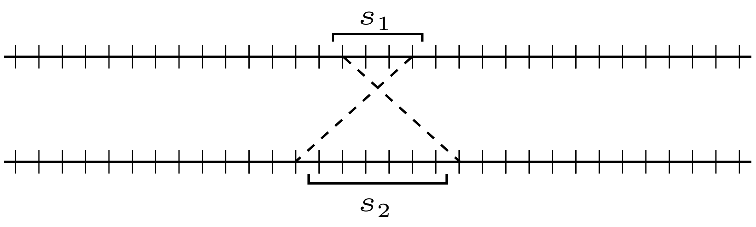

Figure 4 For any s_1 and s_2 travelling in opposite directions, the indicated intervals cannot both contain a starting lockage with the respective ships inside.

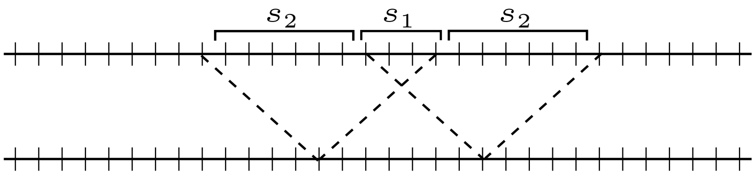

Figure 5 For any s_1 and s_2 travelling in the same direction, the indicated intervals cannot both contain a starting lockage with the respective ships inside.

4 M. Kunst. Organisation of vessel traffic management centres of the future. In *Smart Rivers Conference 2013 Abstract Booklet*, Liege, Belgium, 2013.

5 M. Luy. Ship lock scheduling. `http://sourceforge.net/projects/lockscheduling/`, November 2012.

6 E.R. Petersen and A.J. Taylor. An optimal scheduling system for the Welland Canal. *Transportation Science*, 22:173–185, august 1988.

7 J. Qian and R. Eglese. Finding least fuel emission paths in a network with time-varying speeds. *Networks*, 63(1):96–106, 2014.

8 L. D. Smith, D. C. Sweeney, and J. F. Campbell. Simulation of alternative approaches to relieving congestion at locks in a river transportation system. *Journal of the Operational Research Society*, 60:519–533, 2009.

9 C. Ting and P. Schonfeld. Effects of speed control on tow travel costs. *Journal of waterway, port, coastal, and ocean engineering*, 125(4):203–206, 1999.

10 C. Ting and P. Schonfeld. Control alternatives at a waterway lock. *Journal of waterway, port, coastal, and ocean engineering*, 127(2):89–96, 2001.

11 J. Verstichel, P. De Causmaecker, and G. Vanden Berghe. *The Lock Scheduling Problem*. PhD thesis, KU Leuven, 2013.

A Appendix

A.1 Valid inequalities for the time-indexed model

In the time-indexed model, the constraints that avoid overlap of the lockages contain a single variable $x_{s_1,\ell,t}$ for ship s_1, and a sum over a time interval for ship s_2. A generalization of these constraints is possible by summing over an interval for both ships s_1 and s_2. Graphically, we can represent these constraints as shown in Figures 4 and 5.

For ships s_1 and s_2 travelling in opposite directions, we constrain the movement of s_2 between the latest possible starting time before the start of the interval with s_1 and the earliest possible starting time later than the end of the interval with s_1. We can use a similar approach when s_1 and s_2 are travelling in the same direction. When expressing these new constraints in mathematical notation, we slightly abuse notation and implicitly assume that we do not sum over $x_{s,\ell,t}$ where $t \notin \mathcal{T}$ to avoid formulating similar constraints where the intervals would extend beyond the instance time horizon. The expression for these constraints is then as follows:

$$\sum_{\tau=t}^{t+p} x_{s_1,\ell,\tau} + \sum_{\tau=t+p-P_\ell+1}^{t+P_\ell-1} x_{s_2,\ell,\tau} \leq 1$$
$$\forall \ell \in \mathcal{L}, s_1 \in \mathcal{D}, s_2 \in \mathcal{U}, t \in \mathcal{T}, p \in \{0,\dots,2P_\ell - 2\} \tag{31}$$
$$\sum_{\tau=t}^{t+p} x_{s_1,\ell,\tau} + \sum_{\tau=t-P_\ell+p+1}^{t+P_\ell-1} x_{s_2,\ell,\tau} \leq 1$$
$$\forall \ell \in \mathcal{L}, s_1 \in \mathcal{U}, s_2 \in \mathcal{D}, t \in \mathcal{T}, p \in \{0,\dots,2P_\ell - 2\} \tag{32}$$
$$\sum_{\tau=t}^{t+p} x_{s_1,\ell,\tau} + \sum_{\tau=t+p+1}^{t+2P_\ell-1} x_{s_2,\ell,\tau} + \sum_{\tau=t+2P_\ell}^{t+p+2P_\ell} x_{s_1,\ell,\tau} \leq 1$$
$$\forall \ell \in \mathcal{L}, s_1 \in \mathcal{U}, s_2 \in \mathcal{U}, t \in \mathcal{T}, p \in \{0,\dots,2P_\ell - 1\} \tag{33}$$
$$\sum_{\tau=t}^{t+p} x_{s_1,\ell,\tau} + \sum_{\tau=t+p+1}^{t+2P_\ell-1} x_{s_2,\ell,\tau} + \sum_{\tau=t+2P_\ell}^{t+p+2P_\ell} x_{s_1,\ell,\tau} \leq 1$$
$$\forall \ell \in \mathcal{L}, s_1 \in \mathcal{D}, s_2 \in \mathcal{D}, t \in \mathcal{T}, p \in \{0,\dots,2P_\ell - 1\} \tag{34}$$

Note that for $p = 0$, constraints (32)-(34) imply constraints (8)-(10) of the original IP model. While constraint (32) suffices to prevent overlap of alternating lock movements, we also repeat this constraint for ships moving from downstream to upstream to cut away additional LP-solutions. A tighter LP relaxation can thus be obtained by replacing constraints (8)-(10) by constraints (31)-(34).

A.2 Performance improvement for the lockage-based model

In the lockage-based formulation, the number of variables is largely determined by an upper bound $\mathcal{M}$ on the number of lock movements. Obtaining a lower value for $\mathcal{M}$ could thus significantly improve the performance of the model. We can use the value T that was introduced in the time-index model, i.e. an upper bound on the starting time of the latest non-empty lockage, to obtain a different bound on the number of lockages. Because we may disregard all solutions where a non-empty lockage starts later than T, it follows that each lock $\ell \in \mathcal{L}$ can perform at most $\lfloor T/(2P_\ell) \rfloor + 1$ lockages before violating this requirement. We can thus define an $M_\ell = \min(2S, \lfloor T/(2P_\ell) \rfloor + 1)$ for each lock $\ell \in \mathcal{L}$ and define for each ℓ a set $\mathcal{M}_\ell = \{1, \dots, M_\ell\}$ identifying the different lock movements.

An additional way to further reduce the number of variables is to replace constraints (19)-(21). In the LP-relaxation of the model, fractions of ships may be contained in a lockage. Because lockages must alternate direction, we can impose that either all even-numbered lockages contain only ships in $\mathcal{U}$ and all odd-numbered lockages contain only ships in $\mathcal{D}$, or vice versa. We can do this by enforcing all ships in $\mathcal{U}$ and $\mathcal{D}$ to be restricted to even and odd numbered lockages respectively, while allowing for an additional empty lockage that ends at time $t = 0$. In effect, fixing these variables to zero means eliminating about half the number of $z_{s,l,m}$ variables at the cost of increasing M_ℓ by one and some adjustments to inequalities (22) and (23) discussed below. We can remove constraints (19)-(21) from the formulation after removing the appropriate variables or enforcing the following equalities:

$$z_{s,\ell,m} = 0 \qquad \forall s \in \mathcal{U}, \ell \in \mathcal{L}, m \in \mathcal{M} \cup \{M_\ell + 1\} : \text{ m is even} \tag{35}$$

$$z_{s,\ell,m} = 0 \qquad \forall s \in \mathcal{D}, \ell \in \mathcal{L}, m \in \mathcal{M} \cup \{M_\ell + 1\} : \text{ m is odd} \tag{36}$$

Some care must be taken to allow the first lockages at locks 1 and L to start at time $-2P_1$ and $-2P_L$. Constraints (22) and (23) imply non-negative starting times for all lockages. We can substitute the following for these two constraints:

$$t_{1,m} \geq z_{s,1,m}(A_s + P_1) - P_1 \qquad \forall s \in \mathcal{D}, m \in \mathcal{M} \tag{37}$$

$$t_{L,m} \geq z_{s,L,m}(A_s + P_L) - P_L \qquad \forall s \in \mathcal{U}, m \in \mathcal{M} \tag{38}$$

In addition to these changes, we can note that the lockage based model allows for any number of empty lockages provided that enough time is available to avoid them from overlapping. It is trivial to argue however, that two consecutive empty movements can be removed from any schedule without affecting the solution value. We can thus prune away all solution where two empty lockages are followed by a nonempty lockage. Note that because the number of nonempty lockages in the optimal solution is not known, it is possible however for two or more lockages to be empty if all subsequent lockages are also empty. We can impose this by introducing the following constraints:

$$\sum_{s \in \mathcal{S}} \left(z_{s,\ell,m} + z_{s,\ell,m-1}\right) \geq \frac{1}{S} \sum_{\substack{s \in \mathcal{S} \\ \mu \in \mathcal{M} : \mu > m}} z_{s,\ell,\mu} \quad \forall \ell \in \mathcal{L}, m \in \mathcal{M} \tag{39}$$

The impact on performance of these model adjustments, for both formulations, are discussed in section 6.1.