Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 54-2 (2021) 335–341

Secure Multi-party Co-planning of Barge Departures ⋆

Rie B. Larsen * Ruud Baksteen * Bilge Atasoy *
Rudy R. Negenborn *

* *Department of Maritime and Transport Technology, Delft University of Technology. Delft, The Netherlands (e-mail: R.B.Larsen@tudelft.nl, R.Baksteen-1@student.tudelft.nl, B.Atasoy@tudelft.nl and R.R.Negenborn@tudelft.nl*

Abstract: Container transport requires real-time control and a high degree of cooperation to alleviate disturbances and perform smoothly without unnecessary environmental impact and monetary losses. The involved operators are, however, often reluctant to cooperate as they fear loosing valuable information and autonomy, eventually leading their clients to choose another (cooperating) operator. In this paper we propose a real-time co-planning method, called *Secure Departure Learning* that in real-time lets several truck operators indicate to a barge operator what departure schedules they prefer without revealing any sensitive information. The method uses Paillier encryption and a learning method inspired by Bayesian Optimization in a model predictive control framework. At frequent time intervals a number of potential barge schedules are communicated to the co-planning truck operators. They evaluate their operation cost for each schedule and communicate it to the barge operator encrypted using several public keys. The barge operator computes the encrypted total cost for each schedules, which hereafter is decrypted by several truck operators. The first action of the schedule that results in the lowest total cost is implemented in a model predictive control fashion. Simulated experiments on a realistic, Dutch transport network illustrate that *Secure Departure Learning* is a good alternative for replacing the current method in practice, where barge departures are scheduled ahead of time and only mode-decisions can be updated in real time. *Secure Departure Learning* offers a new perspective on cooperation at the operational level in freight transport where co-planning and information protection can go hand in hand.

Keywords: Cooperative logistics; Multi-modal transport modeling, monitoring and control; Scheduling and optimization of transportation systems; Co-planning; Pailler encryption; Synchromodal transport; Artificial intelligence in transportation

1. INTRODUCTION

Increased real-time cooperation between container transport stakeholders can improve efficiency. This is generally agreed upon among both scholars (Giusti et al., 2019) and practitioners (PWC, 2016). At the operational level it is however difficult to establish close cooperation as many cooperation schemes rely on heavy information sharing or loss of autonomy. In practice, cooperation is often a hierarchy of decisions, where earlier decisions cannot be changed later, thus impeding real-time control. In the following we use the term *co-planning* to describe the, preferably real-time, act of planning transport at the operational level with sharing of limited, consciously chosen information and a clear division of responsibilities and autonomy.

The current literature on cooperation methods at the operational level falls generally into two categories: auctions and distributed optimization. Neither can be classified as co-planning. In auction schemes, shippers and transport providers choose what demand and supply they bring to and bid on at the auction, hence keeping full autonomy. They must however often share sensitive information like pick up and delivery locations before a deal is made (Gansterer et al., 2020). Many auction methods furthermore rely on a neutral, central entity (Li et al., 2015). Often, only very few decisions are re-evaluated or a fixed plan for a future planning period is agreed upon as auctions typically are formulated as computationally complex binary optimization problems. Auction schemes are frequently used in cooperative vehicle routing problems.

Cooperation schemes based on distributed optimization are more common for intermodal transportation. They do often rely on full information sharing and almost infinite communication (Li et al., 2017). Some researchers ensure the interpretability of the methods (Guo, 2020, Chapter 7), but often distributed optimization appears to practitioners as "Black-box" methods giving them a feeling of loosing autonomy. Both static and dynamic/real-time control methods use distributed optimization.

In this paper, we propose a real-time co-planning method, called *Secure Departure Learning* (SDL), that allows a

⋆ This research is supported by the project "Complexity Methods for Predictive Synchromodality" (project 439.16.120) of the Netherlands Organisation for Scientific Research (NWO).

barge operator and multiple truck operators to jointly decide on barge departures. The method ensures the operators' autonomy and individual responsibility by keeping the final decision power local and by only committing to actions when they are to be carried out. The latter is possible under modern concepts, like synchromodal transport, where shippers give full authority over the execution of the transport to the operators. The proposed method builds upon the method presented by Larsen et al. (2020) where one barge and one truck operator cooperate on departure time based on cost information. We expand this method to co-planning with multiple truck operators and ensure no valuable information can be interpreted from the communicated data. The improved data privacy is achieved using Paillier encryption (Paillier, 1999). We furthermore improve the model predictive controller used by the truck operators in Larsen et al. (2020).

To introduce SDL, firstly modelling assumptions, the ideal centralised controller for the problem, and the responsibilities and restrictions of each operator are introduced in Section 2. In Section 3 SDL is presented and discussed and in Section 4 the performance of SDL is shown on simulated experiments. Finally, in Section 5, we provide conclusions and recommendations for further work on co-planning.

2. CONTAINER TRANSPORT CONTROL

If barge and truck operators cooperated without information-exchange limitations, the method used to take decisions in the shared transport system could be formulated centrally. The performance of this method is thus the performance we strive to replicate without sharing sensitive information. In this section the "ideal" centralised controller is thus presented.

Transport at the operational level is constantly facing delays and other disturbances. Synchromodal transport and other modern transport concepts allow transport operators to re-plan container routes freely to accommodate unforeseen events. On the other hand, container deadlines have to be met and each decision incurs a cost. To balance the tradeoff between quick response to changes in the transport system and optimal costs in the long run, the model predictive control (MPC) detailed in Larsen et al. (2020) is used as the centralised transport controller. The formulations have been extended to represent co-planning between a barge operator and multiple truck operators. The MPC has, furthermore, been improved by adding a cost to stacked containers, by encouraging early action, and by including information about due dates that lie outside the prediction horizon.

2.1 Modelling Assumptions

The transport system model used by the centralised controller builds on the following assumptions:

- Network is synchromodal (a-modal bookings and no commitment until departure)
- One barge with sufficient capacity serves two terminals
- Unsatisfied demand is penalized
- Driving trucks loaded and empty cost the same
- One truck can transport one container at a time

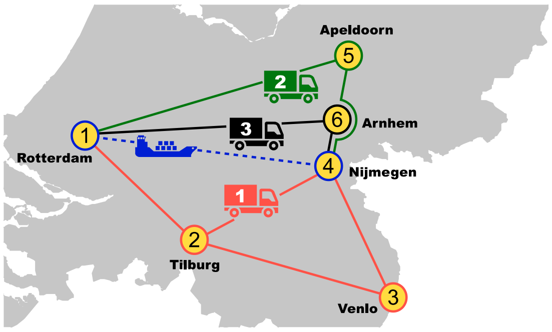

Fig. 1. Transport networks for co-planning with one barge and three truck operators. The networks can overlap.

- Containers and trucks are modelled as integer flows
- Operating hours, terminal capacity, etc., are omitted

The different truck operators in the transport network have different, possibly overlapping, route networks they operate on. Each of these can be described by a graph where the set of nodes, $\mathcal{V}^n$ are the terminals, hubs and way-points used by truck operator n. The two terminals the barge transports between comprise set $\mathcal{B}$ and are part of all truck operators' networks, $\mathcal{B} \in \mathcal{V}^n \ \forall n \in \mathcal{N}$, where $\mathcal{N}$ is the set of truck operators participating in the co-planning. For each node i, the set of nodes connected by barge is $\mathcal{B}_i$ (maximum one element) and the set connected by truck from operator n is $\mathcal{V}_i^n$. The travel time for a truck from operator n from node i to node j is denoted by τ_{ij}^n while the travel time from node i to j by barge is denoted by τ_{ij}^b. Figure 1 shows an example of a transport network.

Containers with the same destination transported by the same truck operator are modelled as one commodity and described as a flow. The state and decision variables are thus vectors, where each element in the vector is representing one commodity. The demand vector $d_i^n(k) \in \mathbb{Z}_{\geq 0}^{m_n}$ describes, therefore, both containers ready to be released into the network and containers needed from the network at node i at time step k. The location in the vector determines what commodity the demand is of, and thus if it has destination at node i or not. Truck operator n transports m_n different commodities. Two mapping vectors, $\psi_i^n \in \{0,1\}^{1\times m_n}$ and $\omega_i^n \in \{0,1\}^{1\times m_n}$ are defined such that $\psi_i^n d_i^n(k)$ is the total number of new containers to be released and $\omega_i^n d_i^n(k)$ is the total number of containers due at node i at time step k.

2.2 Centralized MPC

An MPC reacts to unpredicted events by re-planning the best actions to take at regular time intervals (called time steps). Each plan predicts the effect on the system over the next Tp time steps and optimizes the corresponding actions. Only the actions corresponding to the first time step in the plan, $\kappa = 0$, are implemented before the plan is re-optimized at time step $k+1$. Throughout the paper, k represents the time of the system and $\kappa \in \{0, ..., T_p\}$ describes time in the predicted plan. The feasibility of the current plan is ensured by penalizing unsatisfied demand and by using the current states as initial states of the prediction problem. The latter means that at time step k the predicted number of barges berthed at node i, $z_i^b(\kappa) \in \{0,1\}$, for the initial prediction time step $\kappa = 0$

is the actual number of berthed barges, $\bar{z}^b_{i,k} \in \{0,1\}$. The same is true for the predicted unsatisfied demand, $y^n_i(\kappa)$, the stacked containers, $x^n_i(\kappa)$ and the parked trucks, $z^n_i(\kappa)$, as indicated in (2). The MPC optimization problem is given as follows:

$$\min \sum_{\kappa=0}^{T_p-1} \Bigg(\sum_{n\in\mathcal{N}} \sum_{i\in\mathcal{V}^n} \Big(\gamma x^n_i(\kappa+1) + (1+\xi\kappa) \sum_{j\in\mathcal{V}^n_i} \Big(\pi^n_{ij} v^n_{ij}(\kappa) \Big) + \phi^n y^n_i(\kappa+1) \Big) + \sum_{i\in\mathcal{B}} \sum_{j\in\mathcal{B}_i} \Big(\pi^b_{ij} v^b_{ij}(\kappa) + \sum_{n\in\mathcal{N}} \rho_{ij} \nu^n_{ij}(\kappa) \Big) \Bigg) \tag{1}$$

$$\text{s.t. } z^b_i(0) = \bar{z}^b_{i,k},\ y^n_i(0) = \bar{y}^n_{i,k},\ x^n_i(0) = \bar{x}^n_{i,k},\ z^n_i(0) = \bar{z}^n_{i,k}\ \forall\, i \in \mathcal{V}^n, \forall\, n \in \mathcal{N} \tag{2}$$

$$\text{and } \forall\, i \in \mathcal{V}^n, \forall\, n \in \mathcal{N}, \forall\, \kappa = 0, ..., T_p - 1 :$$

$$z^b_i(\kappa+1) = z^b_i(\kappa) + \sum_{j=\mathcal{B}_i} \left(v^b_{ji}(\kappa - \tau^b_{ji}) - v^b_{ij}(\kappa) \right) \tag{3}$$

$$y^n_i(\kappa+1) = y^n_i(\kappa) - r^n_i(\kappa) - w^n_i(\kappa) + d^n_i(\kappa|k) \tag{4}$$

$$\psi^n_i w^n_i(\kappa) = 0 \tag{5}$$

$$\omega^n_i r^n_i(\kappa) = 0 \tag{6}$$

$$x^n_i(\kappa+1) = x^n_i(\kappa) + \sum_{j=\mathcal{B}_i} \left(\nu^n_{ji}(\kappa - \tau^b_{ji}) - \nu^n_{ij}(\kappa) \right) + r^n_i(\kappa) - w^n_i(\kappa) + \sum_{j\in\mathcal{V}^n_i} \left(u^n_{ji}(\kappa - \tau^n_{ji}) - u^n_{ij}(\kappa) \right) \tag{7}$$

$$z^n_i(\kappa+1) = z^n_i(\kappa) + \sum_{j=\mathcal{V}^n_i} \left(v^n_{ji}(\kappa - \tau^n_{ji}) - v^n_{ij}(\kappa) \right) \tag{8}$$

$$\mathbf{1}^{m_n} u^n_{ij}(\kappa) \le v^n_{ij}(\kappa)\ \forall\, j \in \mathcal{V}^n_i \tag{9}$$

where $\mathbf{1}^{m_n}$ is a row-vector of size m_n of all ones. The binary decision variable $v^b_{ij}(\kappa) = 1$ if it is predicted that the barge departs from node i at time step $k+\kappa$, otherwise $v^b_{ij}(\kappa) = 0$. The due-demand at node i for operator n is satisfied by using $w^n_i(\kappa) \in \mathbb{Z}^{m_n}_{\ge 0}$ containers from the stack, and the released demand is satisfied by adding $r^n_i(\kappa) \in \mathbb{Z}^{m_n}_{\ge 0}$ containers to the stack. Equation (5) and (6) differentiate due and released demand. Containers that depart with the barge at timestep $k+\kappa$ from node i to j are denoted by $\nu^n_{ij}(\kappa) \in \mathbb{Z}^{m_n}_{\ge 0}$ and those departing by truck from operator n are denoted by $u^n_{ij}(\kappa) \in \mathbb{Z}^{m_n}_{\ge 0}$. Trucks from operator n departing from node i towards node j at time step $k+\kappa$ are denoted by $v^n_{ij}(\kappa) \in \mathbb{Z}_{\ge 0}$.

Decisions taken at previous time steps k are known, and can hence be used in the prediction. For example, the number of trucks arriving at node j at time step $k+\kappa$ from node i are $v_{ij}(\kappa - \tau^n_{ij})$, which were decided at time step $k+\kappa-\tau^n_{ij}$ if $k+\kappa-\tau^n_{ij} < k$ and is otherwise a decision variable in the optimization problem at time step k. When necessary, the time of prediction will be specified as, e.g. for truck departures, $v_{ij}(\kappa|k)$.

The demand $d^n_i(\kappa|k)$ is the demand that at time step k is expected to be released at $k+\kappa$ for all $\kappa = 0, ..., T_p-2$. The demand $d^n_i(T_p-1|k)$ is the sum of the expected demand at time step $k+T_p-1$ and the demand due at time step $k+T_p$ to $k+T_p+\theta$. Hereby, the MPC can utilize that due date information often is available when release information is.

The optimization problem minimizes the cost of sailing the unloaded barge, $\phi^b_{ij} v^b_{ij}(\kappa)$, the additional cost of sailing a (partially) loaded barge, $\rho_{ij}\nu^n_{ij}(\kappa)$, the cost of driving the trucks, $\pi^n_{ij} v^n_{ij}(\kappa)$, and the penalty for not satisfying due demand or not accepting released demand on time, $\phi^n y^n_i(\kappa)$. Furthermore, in contrast to Larsen et al. (2020), the objective function also includes a small cost on stacked containers $\gamma x^n_i(\kappa)$ and a small penalty for driving trucks later rather than earlier, $\xi k \pi^n_{ij} v^n_{ij}(\kappa)$.

The transport system can be controlled using this centralized MPC only if all parties are willing to share real-time information on the position of their vehicle fleet and their clients' containers at a commodity level, their (expected) future demand and their cost of operation.

2.3 Co-planning

Both the barge and the truck operators benefit when the barge departures are well aligned with the need for transport of containers by barge. The barge operator avoid sailing empty barges and the truck operators can utilize the cheaper, but slower, barge connection without delivering containers to late. This is the core assumption behind SDL. It is furthermore assumed that the barge operator does not have contact to shippers, and thus does not have other demand than what the truck operators bring. The truck operators do not cooperate among each other, so the demand submitted to a given truck operator must be satisfied by that truck operator, possibly with the use of the barge service. The truck operators are not willing to communicate cost or demand information to the barge operator, neither any information from which this may be inferred. In conclusion, the challenge SDL solves is that all operators have a common goal, but neither operator will share information or let their decision being taken in "black box" fashion.

3. SECURE DEPARTURE LEARNING

In this section SDL is introduced. Firstly, Pailler encryption is described and events are defined. Then follow details of the key steps in SDL and, finally, SDL is presented.

3.1 Preliminaries and definitions

SDL uses Paillier encryption to ensure no information can be interpreted by any one of the cooperating operators. In the following, the encryption method and its main property is described. For more information, we refer to Paillier (1999).

Definition 1. The cost $J \in \mathbb{Z}$ is encrypted using encryption scheme n by

$$E_n(J) = {g_n}^J h^{o_n} \bmod o_n^2, \tag{10}$$

where $o_n = p_n q_n > J$, p_n and q_n being large prime numbers, and $0 < h < o_n$ is a random number. g_n is a random element from the set Ψ_n. Ψ_n is the disjoint union of all $\Psi_n(a)$, which are the sets of elements of order $o_n a$ for $a = 1, ..., \lambda(o_n)$. $\lambda(o_n)$ is the Carmichael's function taken on o_n. The encrypted number $a < o_n^2$ is decrypted using encryption scheme n by

$$D_n(a) = \frac{L(a^{\lambda(o_n)} \bmod o_n^2)}{L(g^{\lambda(o_n)} \bmod o_n^2)} \bmod o_n, \tag{11}$$

where $L(\chi) = \frac{\chi-1}{o_n}$.

Definition 2. (o_n, g_n) is the *public key* for encryption scheme n. (p_n, q_n) or equivalently $\lambda(o_n)$ is the *private key* for encryption scheme n.

If an operator knows the public key, he can encrypt data using scheme n. If he knows both the public and the private key, he can encrypt and decrypt data.

Fact 1. Paillier encryption is additively homomorphic, such that $\forall\, J_1,\, J_2 \in \mathbb{Z}_{<o_n}$ and $\sigma \in \mathbb{Z}_{>0}$:

$$D_n(E_n(J_1)E_n(J_2) \bmod o_n^2) = J_1 + J_2 \bmod o_n \qquad (12)$$

$$D_n(E_n(J_1)^{\sigma} \bmod o_n^2) = \sigma J_1 \bmod o_n \qquad (13)$$

SDL uses learning based on ideas from Bayesian optimization to decide on departure schedules. To decrease the complexity of the learning problem, the sequences of departures are described as events. For more information on events, we refer to Larsen et al. (2020).

Definition 3. An event $e(k)$ is the sum of two departure sequences $v^b_{ij}(\kappa)$ and $v^b_{ji}(\kappa)\ \forall\, \kappa = 0, ..., T_p - 1,\ \{i, j\} = \mathcal{B}$. The set of feasible events $\mathcal{E}(k)$ contains all the events that are feasible at time step k, i.e. satisfies $z^b_i(0) = \bar{z}^b_{i,k}$ and (3) $\forall\, \kappa = 0, ..., T_p - 1$.

An example of the construction of an event is:

$$\begin{array}{lcl} [v^b_{ij}(0) \cdots v^b_{ij}(T_p-1)] & = & [\,0\ 0\ 1\ 0\ 0\ 0\,] \\ [v^b_{ji}(0) \cdots v^b_{ji}(T_p-1)] & = & [\,1\ 0\ 0\ 0\ 1\ 0\,] \\ e & = & [\,1\ 0\ 1\ 0\ 1\ 0\,] \end{array}$$

Conjecture 2. When the current location of the barge is known, i.e. $\bar{z}^b_{i,k}$ and $v^b_{ij}(0|k-a),\ \forall\, a \in \{\tau_{ji}, ..., 2\tau_{ji}\}$ $\forall\, i \in \mathcal{B},\ j \in \mathcal{B}_i$ are known. Then the sequences of departures $v^b_{ij}(\kappa|k)$ and $v^b_{ji}(\kappa|k)\ \forall\, \kappa = 0, ..., T_p - 1,\ \{i, j\} = \mathcal{B}$ can be inferred error-free from an event e using

$$v^b_{ij}(\kappa|k) \leq \bar{z}^b_{i,k} + \sum_{a=\tau^b_{ji}-\kappa}^{\tau^b_{ji}} v^b_{ji}(0|k-a)\ \forall\, \kappa \leq \tau^b_{ji}. \qquad (14)$$

Definition 4. A timeless event e^{∞} is an event e seen over an infinite time horizon, i.e. $e^{\infty} = \left[\mathbf{0}_{1:k}\ e\ \mathbf{0}_{k+T_p:\infty}\right]$, where $\mathbf{0}_{a:b} = \{0\}^{1\times b-a}$ is a zero-vector of suitable size. Note that different events e at different time steps k can have the same timeless event.

Definition 5. A neighbouring event is an event e_2 that differs from e_1 by only one departure time. This departure is shifted one time step either earlier or later and as such e_1 and e_2 are considered neighbours. The set of neighbouring events for event $e_1 \in \mathcal{E}(k)$ is $\mathcal{H}_{e_1^{\infty}}(k) = \{e_2 \in \mathcal{E}(k) \setminus \{e_1\} | v_2(\kappa) = v_1(\kappa) \forall \kappa \in \{0, ..., T_p - 1\} \setminus \{a, a+1\}$, where $v_2(a) = v_1(a+1)$ and $v_1(a) = v_2(a+1)\}$, where $v_a(\kappa) = v^b_{ij}(\kappa) + v^b_{ji}(\kappa),\ \{i, j\} = \mathcal{B}$ for event e_a.

Below are examples of timeless events and neighbours:

$$\begin{array}{lcl} e_{a1} \in \mathcal{E}(k) & = & [\,0\ 1\ 0\ 0\ 1\ 0\,] \\ e_{a2} \in \mathcal{E}(k+1) & = & \quad[\,1\ 0\ 0\ 1\ 0\ 0\,] \\ e^{\infty}_{a1} = e^{\infty}_{a2} = e^{\infty}_{a} & = & 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \\ e_b \in \mathcal{H}_{e^{\infty}_a}(k) & = & [\,1\ 0\ 0\ 0\ 1\ 0\,] \\ e_{c1} \in \mathcal{H}_{e^{\infty}_a}(k) & = & [\,0\ 0\ 1\ 0\ 1\ 0\,] \\ e_{c2} \in \mathcal{H}_{e^{\infty}_a}(k+1) & = & \quad[\,0\ 1\ 0\ 1\ 0\ 0\,] \end{array}$$

3.2 Schedule evaluation

The truck operators do not have authority to decide on when the barge departs. They can optimize the movements of containers and trucks in accordance with the schedule decided by the barge operator. From a truck operator's point of view, each event e at time k thus has a cost of $J^n(e, k)$, which can be computed as:

$$J^n(e,k) = \min \sum_{\kappa=0}^{T_p-1} \sum_{n\in\mathcal{N}} \sum_{i\in\mathcal{V}^n} \Bigg((1+\xi k) \sum_{j\in\mathcal{V}^n_i} \pi^n_{ij} v^n_{ij}(\kappa) + \sum_{j\in\mathcal{B}_i} \rho_{ij} \nu^n_{ij}(\kappa) + \gamma x^n_i(\kappa) + \phi^n y^n_i(\kappa) \Bigg) \qquad (15)$$

$$\text{s.t. } y^n_i(0) = \bar{y}^n_{i,k},\ x^n_i(0) = \bar{x}^n_{i,k},\ z^n_i(0) = \bar{z}^n_{i,k}\ \forall\, i \in \mathcal{V}^n \qquad (16)$$

$$(4) - (9)\ \forall\, i \in \mathcal{V}^n,\ \forall\, \kappa = 0, ..., T_p - 1. \qquad (17)$$

The truck operators are willing to give encrypted information about the cost associated with η events at each time step k. The set of evaluated events is denoted by $\mathcal{I}(k)$.

3.3 Favourable schedules

The barge operator is interested in minimizing the cost of transport for all the cooperating operators, i.e. at every time step k finds the schedule that minimizes (1)-(9). Since the barge operator only has authority to decide upon departures, he needs cost information from the truck operators to find the cost of an event.

Definition 6. The fitness of the timeless event e^{∞} for event $e \in \mathcal{E}(k)$ at time step k is

$$F_{e^{\infty}}(k) = \sum_{n\in\mathcal{N}} J^n(e,k) + \sum_{\kappa=0}^{T_p-1} \sum_{i\in\mathcal{B}} \sum_{j\in\mathcal{B}_i} \pi^b_{ij} v^b_{ij}(\kappa), \qquad (18)$$

where the departure sequences $v^b_{ij}(\kappa)$ and $v^b_{ji}(\kappa)\ \forall\, \kappa = 0, ..., T_p - 1,\ \{i, j\} = \mathcal{B}$ are inferred from e using (14).

Since the transport system is controlled in an MPC fashion (with receding horizon), only the actions corresponding to $\kappa = 0$ are implemented before a new schedule is made. It is therefore assumed that performance information about a schedule at time step k can help estimating the performance of similar schedules at time step $k+1$. Specifically, it is assumed that events sharing the same timeless event or being neighbours incur similar costs.

Definition 7. The set of events that are feasible at both time step k and $k+1$ is $\mathcal{H}^{+}_{e^{\infty}}(k) = \mathcal{H}_{e^{\infty}}(k) \cup \mathcal{H}_{e^{\infty}}(k+1)$ and contains $\eta_{e^{\infty}}(k)$ events.

The estimated fitness of the event $e \in \mathcal{E}(k+1)$ is

$$\tilde{F}_{e^{\infty}}(k+1) = \alpha \tilde{F}_{e^{\infty}}(k) + \frac{1-\alpha}{\eta_{e^{\infty}}(k)} \sum_{a\in\mathcal{H}^{+}_{e^{\infty}}(k)} \tilde{F}_{a^{\infty}}(k), \qquad (19)$$

if $e_{\infty} \in \mathcal{E}(k)$; otherwise $\tilde{F}_{e^{\infty}}(k+1) = \max_{a\in\mathcal{I}(k)} \tilde{F}_{a^{\infty}}(k)$. $0 \leq \alpha \leq 1$.

If an event's timeless event is evaluated recently, it is more likely to estimate the fitness well. For each event a measure of uncertainty is defined.

Definition 8. The uncertainty value for event $e \in \mathcal{E}(k+1)$ is

Algorithm 1 Barge operator's strategy for deciding $\mathcal{I}(k)$

```
1:  input $\tilde{F}_{e^\infty}(k)$, $\tilde{s}_{e^\infty}(k)$, $\mathcal{E}(k)$
2:  return $\mathcal{I}(k)$ with $\eta$ unique events
3:  $\mathcal{I}(k) = \emptyset$
4:  for $i \leftarrow 1$ to $floor(\eta/6)$ do
5:      $\mathcal{I}(k) = \mathcal{I}(k) \cup \arg\min_{e \in \mathcal{E}(k)\setminus\mathcal{I}(k)} \tilde{F}_{e^\infty}(k) + \tilde{s}_{e^\infty}(k)$
6:      $\mathcal{I}(k) = \mathcal{I}(k) \cup \arg\min_{e \in \mathcal{E}(k)\setminus\mathcal{I}(k)} \tilde{F}_{e^\infty}(k)$
7:      $\mathcal{I}(k) = \mathcal{I}(k) \cup \arg\max_{e \in \mathcal{E}(k)\setminus\mathcal{I}(k)} \tilde{s}_{e^\infty}(k)$
8:      $\mathcal{I}(k) = \mathcal{I}(k) \cup \arg\min_{e \in \mathcal{E}(k)\setminus\mathcal{I}(k)} \tilde{F}_{e^\infty}(k) - \tilde{s}_{e^\infty}(k)$
9:      for $j \leftarrow 1$ to 2 do
10:         $\mathcal{I}(k) = \mathcal{I}(k) \cup \mathrm{rand}\,(e \in \mathcal{E}(k) \setminus \mathcal{I}(k))$
11:     end for
12: end for
13: for $i \leftarrow floor(\eta/6)6$ to $\eta$ do
14:     $\mathcal{I}(k) = \mathcal{I}(k) \cup \mathrm{rand}\,(e \in \mathcal{E}(k) \setminus \mathcal{I}(k))$
15: end for
```

$$\tilde{s}_{e^\infty}(k+1) = (\alpha+\beta)\tilde{s}_{e^\infty}(k) + \frac{1-\alpha}{\eta_{e^\infty}(k)} \sum_{a \in \mathcal{H}^+_{e^\infty}(k)} \tilde{s}_{a^\infty}(k), \tag{20}$$

if $e_\infty \in \mathcal{E}(k)$; otherwise $\tilde{s}_{e^\infty}(k+1) = s_{\text{new}}$. s_{new} is a large number. The factor $\beta \in \mathbb{R}_{\geq 0}$ increases the uncertainty of earlier estimates.

The core idea in SDL is that the barge operator at every time step k consciously chooses η events that the truck operators evaluate and provide feedback on. It is important that the set of events to be evaluated $\mathcal{I}(k)$ contains events that both exploit the information known from previous time steps and explore the search space. This balance is achieved by using Algorithm 1 from Larsen et al. (2020).

3.4 Secure departure learning

In SDL the barge and the truck operators communicate forth and back three times before it is decided if the barge should depart right now or not. The first time, the barge operator sends the truck operators a set of schedules he believes will either be close to optimal or valuable for future estimations (computed using Algorithm 1). The truck operators compute individually how much the total cost of transport would be if each of the schedules were implemented in full (using (15)-(17)). Each truck operators has the public encryption key from all of the participating truck operators. Truck operator n encrypts the cost for each schedule with all the available public keys individually ($E_a(J^n(e,k)\,\forall\, e \in \mathcal{I}(k), \forall\, a \in \mathcal{N})$ and communicates them back to the barge operator.

The barge operator then computes his cost for each schedule, encrypts it and adds it to the received costs using (12) (resulting in $E_n(F_{e^\infty}(k))\,\forall\, n \in \mathcal{N}$). The barge operator now scales the encrypted total cost with an integer only known to him $\sigma_n(k)$. This integer varies over time and for each event to ensure the truck operators can not infer the true total cost. The scaled total cost encrypted with operator n's key ($D_n(E_n(F_{e^\infty}(k))^{\sigma_n(k)} \bmod o_n^2)$) is sent to operator n, who decrypts it using his private key and returns $\sigma_n(k)F_{e^\infty}(k)$.

The best schedule can now be found by the barge operator after de-scaling the received costs. This schedule is communicated to all truck operators, who ensure they plan in accordance. All operators implement the first decisions in their plans and the barge operator updates the fitness estimates and uncertainty values to prepare to repeat the process at the next time step. All action and communication steps of SDL are shown in Figure 2.

The total cost is scaled by the barge operator to hide the total cost of the cooperation from the competing truck operators. It is thus only the barge operator who knows the true total cost. The public keys of everybody must be known by all operators such that the barge operator can sum the encrypted costs from the truck operators and add his own encrypted cost. If it is undesirable that the truck operators know how many operators are co-planning, a subset of the keys can be distributed. It is in that case important that sufficiently many keys are known by all operators such that the barge operator can compare the incoming results to ensure they have been decrypted truthfully. The barge operator must not have access to any of the private keys, since this will enable him to decrypt individual truck operators' costs.

If it is undesirable that the barge operator knows the true total cost, one solution is that the truck operators modify the returned costs by scaling with a constant factor $\delta \in \mathbb{R}$. To ensure the barge operator cannot infer the scaling factor δ easily, the truck operators can choose $\delta \geq 1$ and add a number smaller than an agreed minimum value for $\sigma_n(k)$. This way the barge operator cannot simply find the δ through the greatest common divisor of the received costs, while the error in the precision of the total cost will not be more severe than the error introduced by using integer values for the encryption.

Using Paillier encryption, the costs will be rounded to integer values. This is assumed acceptable as the decimals of transport cost often can be disregarded and if deemed important communication precision can be agreed upon, e.g., using cents instead of euros.

SDL relies on estimates of the fitness of each feasible event to learn good departure schedules. This requires a large memory and computations over many entities. The method is however easily parallelizable both for truck operators, where each event can be evaluated independently, and for the barge operator, where Algorithm 1 and the updates (19) and (20) can be parallelized. See Larsen et al. (2020) for further information.

4. SIMULATION EXPERIMENTS

The performance of SDL is in this section compared to that of the centrallized MPC, (1)-(9), and the performance of a fixed schedule. We refer to these as the *central* and the *fixed* methods. The fixed method represents the current practice where a barge departs at the beginning of the simulation from Nijmegen and departs hereafter every 6,5 hours alternating between this terminal and Rotterdam. Each truck operator knows this schedule and employs their own MPC using (15)-(17).

The experiments were simulated in Matlab formulated with Yalmip (Löfberg, 2004) and solved by Gurobi. Ex-

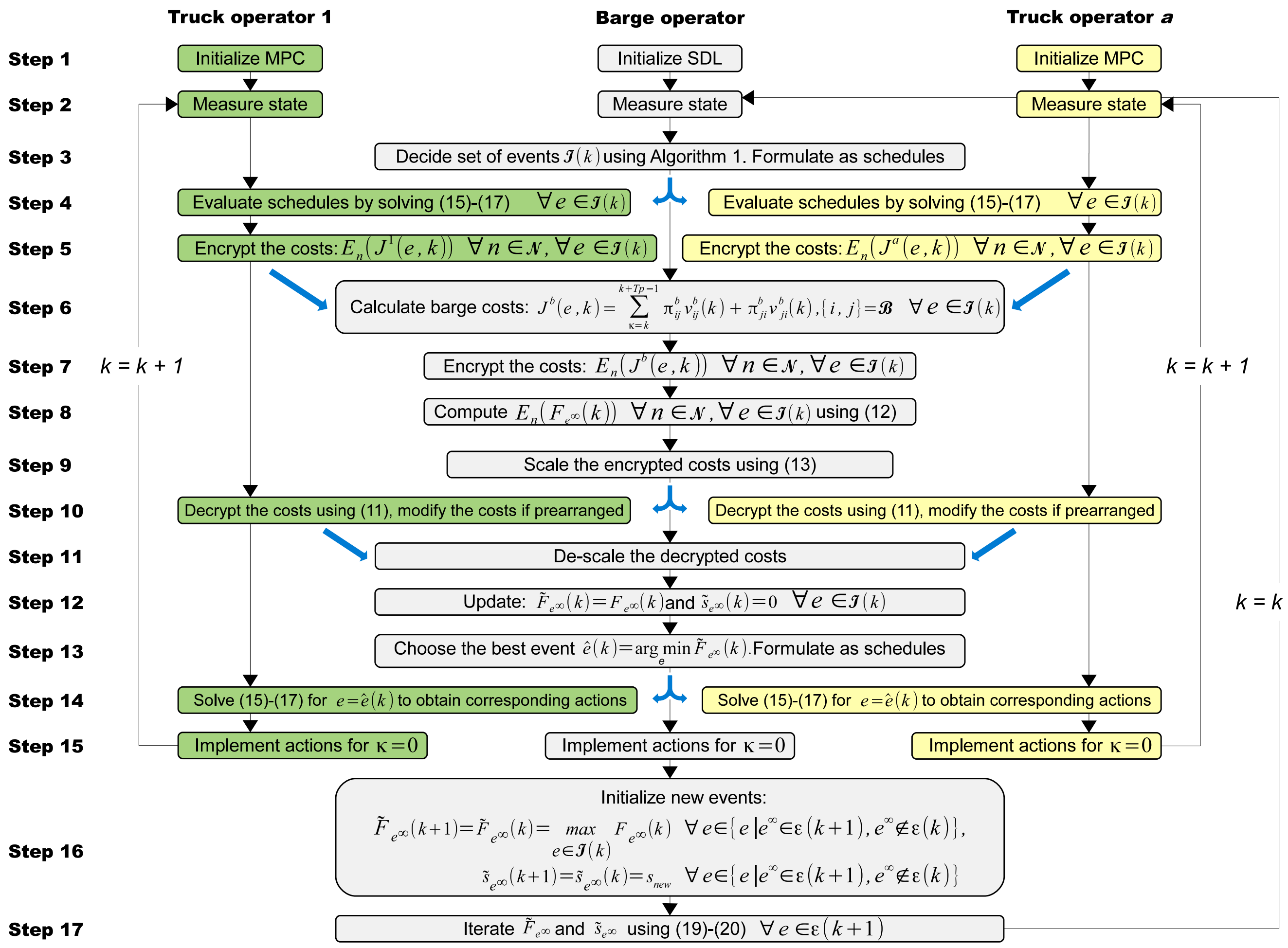
Truck operator 1
Barge operator
Truck operator a
Step 1
Initialize MPC
Initialize SDL
Initialize MPC
Step 2
Measure state
Measure state
Measure state
Step 3
Decide set of events $\mathcal{I}(k)$ using Algorithm 1. Formulate as schedules
Step 4
Evaluate schedules by solving (15)-(17) $\forall e \in \mathcal{I}(k)$
Evaluate schedules by solving (15)-(17) $\forall e \in \mathcal{I}(k)$
Step 5
Encrypt the costs: $E_n(J^1(e,k))$ $\forall n \in \mathcal{N}, \forall e \in \mathcal{I}(k)$
Encrypt the costs: $E_n(J^a(e,k))$ $\forall n \in \mathcal{N}, \forall e \in \mathcal{I}(k)$
Step 6
Calculate barge costs: $J^b(e,k) = \sum_{\kappa=k}^{k+Tp-1} \pi^b_{ij} v^b_{ij}(k) + \pi^b_{ji} v^b_{ji}(k), \{i,j\} = \mathcal{B}$ $\forall e \in \mathcal{I}(k)$
Step 7
$k = k+1$
Encrypt the costs: $E_n(J^b(e,k))$ $\forall n \in \mathcal{N}, \forall e \in \mathcal{I}(k)$
$k = k+1$
Step 8
Compute $E_n(F_{e^\infty}(k))$ $\forall n \in \mathcal{N}, \forall e \in \mathcal{I}(k)$ using (12)
Step 9
Scale the encrypted costs using (13)
Step 10
Decrypt the costs using (11), modify the costs if prearranged
Decrypt the costs using (11), modify the costs if prearranged
Step 11
De-scale the decrypted costs
Step 12
Update: $\tilde{F}_{e^\infty}(k) = F_{e^\infty}(k)$ and $\tilde{s}_{e^\infty}(k) = 0$ $\forall e \in \mathcal{I}(k)$
$k = k+1$
Step 13
Choose the best event $\hat{e}(k) = \arg\min_e \tilde{F}_{e^\infty}(k)$. Formulate as schedules
Step 14
Solve (15)-(17) for $e = \hat{e}(k)$ to obtain corresponding actions
Solve (15)-(17) for $e = \hat{e}(k)$ to obtain corresponding actions
Step 15
Implement actions for $\kappa = 0$
Implement actions for $\kappa = 0$
Implement actions for $\kappa = 0$
Step 16
Initialize new events:
$\tilde{F}_{e^\infty}(k+1) = \tilde{F}_{e^\infty}(k) = \max_{e \in \mathcal{I}(k)} F_{e^\infty}(k)$ $\forall e \in \{e \mid e^\infty \in \varepsilon(k+1), e^\infty \notin \varepsilon(k)\}$,
$\tilde{s}_{e^\infty}(k+1) = \tilde{s}_{e^\infty}(k) = s_{new}$ $\forall e \in \{e \mid e^\infty \in \varepsilon(k+1), e^\infty \notin \varepsilon(k)\}$
Step 17
Iterate $\tilde{F}_{e^\infty}$ and $\tilde{s}_{e^\infty}$ using (19)-(20) $\forall e \in \varepsilon(k+1)$

Fig. 2. Action (black arrows) and communication (bold, blue arrows) flow for SDL.

periments with Paillier encryption were performed using the toolbox by D'Errico (2020) to ensure precision of the large integers. However, as encryption does not impact the performance, the shown experiments were performed in plaintext.

For the experiments, a period of 5 days of transport on the network shown in Figure 1 is considered. The truck networks are not overlapping except for the barge terminal nodes. This is not a limitation of DSL, but is chosen to simplify the presentation of the experiments. Time was discretized with intervals of 15 min, giving a total simulation length of 480 time steps. All three methods used prediction horizon $T_p = 70$, $\theta = 70$ and the parameters shown in Table 1. The same table shows the costs and the non-zero initial states. SDL was initialized with $\tilde{F}_{e^\infty}(0) = 10000$ and $\tilde{s}_{e^\infty}(0) = s_{\text{new}} = 5000$, and used $\alpha = 0.8$, $\beta = 0.1$ and $\eta = 30$.

The experiments were conducted with both stable demand and demand with peaks. All methods know the exact demand for the prediction horizon T_p at every time step. For both profiles, between 0 and 3 containers of each commodity are released at all time steps at all terminals except Nijmegen. They are each assigned a leadtime of minimum 70 time steps and added to the due-demand at the corresponding time step at the corresponding terminal. No containers are due at Nijmegen.

The demand with peaks simulate how a hinterland network typically is stressed by the large amounts of containers arriving and departing by ships in Rotterdam. The stable demand forms the basis of this demand profile, but in addition it adds between 0 and 4 release containers at all nodes (except Nijmegen) at time steps k =[14:20,60:65,100:104,150:162,199:208,250:258,301:311, 344:350,400:408,430:436]. Their due-time is computed as for the stable demand.

4.1 Results

SDL performs better than the fixed method but, as expected, not as well as the centralized MPC for both demand profiles. This is as expected, since more cooperation

Table 1. Network parameters, costs (in €) and non-zero initial states used in the experiments

$\tau^b_{14} = \tau^b_{41} = 24$	$\pi^b_{14} = \pi^b_{41} = 60$	$\bar{z}^b_{4,0} = 1$	$\rho_{14} = 28.18$
$\tau^1_{12} = \tau^1_{21} = 6$	$\pi^1_{12} = \pi^1_{21} = 44.26$	$\bar{z}^1_{1,0} = 33$	$\rho_{41} = 28.18$
$\tau^1_{23} = \tau^1_{32} = 6$	$\pi^1_{23} = \pi^1_{32} = 45.98$	$\bar{z}^1_{2,0} = 33$	$\gamma = 0.0001$
$\tau^1_{34} = \tau^1_{43} = 5$	$\pi^1_{34} = \pi^1_{43} = 37.37$	$\bar{z}^1_{3,0} = 33$	$\xi = 0.001$
$\tau^1_{24} = \tau^1_{42} = 5$	$\pi^1_{24} = \pi^1_{42} = 39.10$		$\phi^1 = 1000$
$\tau^2_{15} = \tau^2_{51} = 9$	$\pi^2_{15} = \pi^2_{51} = 63.19$	$\bar{z}^2_{1,0} = 35$	$\phi^2 = 1000$
$\tau^2_{45} = \tau^2_{54} = 4$	$\pi^2_{45} = \pi^2_{54} = 33.93$		$\phi^3 = 1000$
$\tau^3_{16} = \tau^3_{61} = 8$	$\pi^3_{16} = \pi^3_{61} = 54.59$	$\bar{z}^3_{6,0} = 30$	
$\tau^3_{46} = \tau^3_{61} = 3$	$\pi^3_{46} = \pi^3_{64} = 21.88$		

Table 2. Simulation results

Demand	Control method	Realised cost (1,000€)	Barge departures	Containers per barge (average)	Truck departures	Empty truck departures	Truck travel distance (1,000 km)	CO2 emission (tonne)
Stable	SDL	542	14	69	10,800	328	1,022	959
	fixed	548	19	59	10,962	380	1,021	968
	central	535	5	116	10,701	227	1,038	908
Peaks	SDL	685	12	333	13,156	761	1,087	1074
	fixed	696	19	245	13,485	1065	1,050	1104
	central	679	9	408	13,181	715	1,095	1035

enables better utilization of the vehicles. This is especially clear from the number of barge departures, where SDL plans for significantly fewer departures than what is in the fixed schedule but does not decrease as much as the centralized MPC. When the demand is stable, the performance of SDL is closer to that of the fixed method. On the other hand, it is closer to that of the centralized MPC when the demand has peaks. This is explained by the increased importance of aligning barge departures with demand when many containers have the same transport need. The same pattern is observed for truck departures, especially in the number of empty departures. The trucks are however used on longer routes more frequently for the centralized MPC, which results in an increase in the total distance driven by trucks. The additional cost and CO2 from extra barge departures are, however, lower than the savings achieved by using fewer, better utilized barges, such that the total cost of transport and the resulting CO2 emissions decrease. All three methods satisfy the demand in time.

The CO2 emissions for the transports are computed using the guidelines by ECTA (2011). It is not directly considered by any of the methods (it is not part of the objective functions). This shows that it is important to create methods that are realistic for transport operators to use for co-planning - not only for the cost and efficiency, but also for the environment.

5. CONCLUSIONS AND FUTURE RESEARCH

The presented method, *Secure Departure Learning* (SDL), lets a barge operator co-plan departure times with multiple truck operators without revealing any sensitive information between the cooperating parties. This is achieved using Paillier encryption and a model predictive control based method integrated with learning using ideas from Bayesian optimization. The method can be used without either party giving up autonomy or risk loosing information-based advantages and can as such easier replace the current practice, where the barge operator fixes the schedule ahead of time.

Compared to using a fixed schedule, SDL increases the utilization rate of especially barges but also trucks. A centralized decision maker outperforms SDL as expected, but is almost impossible to implement in real world applications due to lack of interested participants. Increased utilization rates improve not only the total transport cost, but also the CO2 emissions. In conclusion, SDL is a proof of concept for the idea that it is possible for transport operators to co-plan without revealing any sensitive information.

Future research on SDL should include experiments on multiple cases to strengthen the conclusions. The method could furthermore be improved by considering the environmental impact directly and including opening hours and similar constraints in the mathematical model. SDL does not consider the environmental impact directly, nor does it consider opening hours and similar constraints. An extension of the method towards more realistic assumptions and direct consideration of environmental impact are interesting future research directions. An important assumption to overcome before SDL can be deployed for practitioners is the unlimited barge capacity. This may only be overcome by the use of more advanced encryption techniques. Further investigation into the impact of SDL's meta-parameters would also improve the understanding of the method and the possibilities for extension towards more complex co-planning with multiple barges that decide both departure times and routing.

REFERENCES

D'Errico, J. (2020). Variable precision integer arithmetic. MATLAB Central File Exchange.

ECTA (2011). Guidelines for measuring and managing CO2 emission from freight transport operations.

Gansterer, M., Hartl, R.F., and Savelsbergh, M. (2020). The value of information in auction-based carrier collaborations. *International Journal of Production Economics*, 221, 107485.

Giusti, R., Manerba, D., Bruno, G., and Tadei, R. (2019). Synchromodal logistics: An overview of critical success factors, enabling technologies, and open research issues. *Transportation Research Part E: Logistics and Transportation Review*, 129, 92–110.

Guo, W. (2020). *Dynamic, Stochastic, and Coordinated Optimization for Synchromodal Matching Platforms.* Ph.D. thesis, Delft University of Technology.

Larsen, R.B., Atasoy, B., and Negenborn, R.R. (2020). Learning-based co-planning for improved container, barge and truck routing. In E. Lalla-Ruiz, M. Mes, and S. Voß (eds.), *Computational Logistics*, 476–491.

Li, J., Rong, G., and Feng, Y. (2015). Request selection and exchange approach for carrier collaboration based on auction of a single request. *Transportation Research Part E: Logistics and Transportation Review*, 84, 23–39.

Li, L., Negenborn, R.R., and De Schutter, B. (2017). Distributed model predictive control for cooperative synchromodal freight transport. *Transportation Research Part E: Logistics and Transportation Review*, 105, 240–260.

Löfberg, J. (2004). YALMIP : A toolbox for modeling and optimization in matlab. In *Proc. of Int. Symposium on Computer Aided Control System Design.*

Paillier, P. (1999). Public-key cryptosystems based on composite degree residuosity classes. In J. Stern (ed.), *Advances in Cryptology– EUROCRYPT*, 223–238.

PWC (2016). Shifting Patterns- The Future of the Logistics Industry. Technical report, PWC.