Transportation Research Part C 132 (2021) 103383

Contents lists available at ScienceDirect

Transportation Research Part C

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

Integrating stochastic programs and decision trees in capacitated barge planning with uncertain container arrivals

Volkan Gumuskaya [a,*], Willem van Jaarsveld [a], Remco Dijkman [a], Paul Grefen [a], Albert Veenstra [b]

[a] *School of Industrial Engineering, Eindhoven University of Technology, The Netherlands*
[b] *Department of Technology and Operations Management, Rotterdam School of Management, The Netherlands*

ARTICLE INFO

Keywords:
Hinterland transport
Stochastic program
Machine learning
Barge planning
Decision tree
Simulation

ABSTRACT

In this paper, we propose an approach that combines optimization techniques with machine learning to improve capacitated barge planning with uncertain container arrivals. The main idea is to use the predictions of a decision tree in the scenario generation of a 2-stage stochastic program to plan the barge calls (i.e. visits) of a barge operator. The predictions of container arrivals help to generate more accurate scenarios, which in turn leads to more informed decisions and less costs. The approach is tested with an iterative method of periodic planning and simulation for a one year duration so that the long term performance is evaluated. A computational experiment is conducted using the historical data of an inland terminal and the Port of Rotterdam. The results show that the proposed approach improves total costs up to 2.07% over the traditional stochastic approach, and up to 4.57% over the current method used in industry.

1. Introduction

Barge transport is potentially a cheaper and more environmental friendly alternative to road transport due to economies of scale facilitated by high capacity vehicles. Barges are an effective mode in hinterland transport networks because of the high volumes of export and import flows between a deep sea port and inland terminals. Indeed, 38% of the container flow between the Port of Rotterdam and the hinterland is transported by barges.[1] The vision of the Port of Rotterdam is to increase the share of barge transport even more, up to 45% by 2030. Dealing with uncertainty is an important challenge in achieving this vision. Uncertain events such as container arrival delays lead to operational problems and additional costs, which in turn hinder the competitive power of barge transport against road transport (Gumuskaya et al., 2020b).

Planning under uncertainty in freight transport has typically been addressed by combinatorial optimization techniques such as stochastic programming, Markov decision processes (MDP) and robust optimization. To further alleviate the negative effects of uncertainty, data can be used to support the decision making process. By providing more accurate information, such as better forecasts, more informed and consequently better decisions can be made. Recently, due to the success of Machine Learning (ML) methods in other fields, combining combinatorial optimization with ML has received attention from academia. These efforts are often aimed towards making the existing algorithms faster. Examples of such efforts include improving the branch and bound technique using decision trees (Lodi and Zarpellon, 2017) and using mixed integer optimization for classification (Bertsimas et al., 2011).

* Corresponding author.
E-mail address: v.gumuskaya@tue.nl (V. Gumuskaya).

[1] https://www.portofrotterdam.com/en/doing-business/logistics/connections/intermodal-transportation/inland-shipping/optimising-inland.

https://doi.org/10.1016/j.trc.2021.103383
Received 2 December 2020; Received in revised form 6 September 2021; Accepted 7 September 2021
Available online 29 September 2021

Another common approach is to solve combinatorial problems that cannot be solved in polynomial time by machine learning. For example, the Traveling Salesman Problem (TSP) has been solved by neural networks and reinforcement learning (Vinyals et al., 2015; Bello et al., 2016), and van Riessen et al. (2016) use decision trees to solve the container allocation problem, which are typical combinatorial optimization problems.

In this paper, we propose an approach that integrates machine learning with optimization. In contrast to the previous work we mentioned, our approach focuses on using machine learning to improve the *solution quality* (i.e. the objective function of an optimization problem), instead of improving the solution time. More specifically, we propose using the predictions of a decision tree to improve the scenario generation process of a 2-stage stochastic program, solved by Sample Average Approximation (SAA). The decision tree is trained on historical data, which captures the patterns regarding the uncertainty. The predictions on the uncertain factors help to generate more accurate scenarios and improve decision making. Decision trees are typically used to make class predictions, i.e. point forecasts for the uncertain variables. Using directly the class predictions results in a form of deterministic approach. Instead, we use the probabilistic predictions that are implicitly available from the training of the decision tree to generate scenarios. As such we employ a stochastic perspective, and distinguish between predictions made with high confidence from those made with less confidence. Regarding the uncertain variables, the former yields more homogeneous scenarios favoring the predicted class, while the latter yields more heterogeneous scenarios. This enables hedging the risk of incorrect predictions, and avoids 'putting all eggs in one basket'.

The proposed approach is applied to capacitated barge planning between an inland terminal and multiple container terminals in a deep sea port, with a context similar to that of Gumuskaya et al. (2020a). The containers have an expected arrival date but delays (in days) may occur due to deep sea vessel arrivals or preparation of cargo, which is the uncertain factor. First, a decision tree is trained on historical data consisting of container specific information and observed delays. The decision tree predicts the probabilities of arrival delays, which are then used in the periodic barge planning. Each period consists of multiple days, and the barge plan is made using a 2-stage stochastic program, of which the scenario generation process is executed using the probabilistic predictions. The barge planning decisions are then evaluated with a simulation in order to determine the actual costs. In simulation, actual delays are randomly sampled from historical data disjunct from the training set of the decision tree. The planning–simulation procedure is repeated iteratively to represent one year of operations in order to check the long term performance.

The proposed approach is compared with four benchmark methods: (1) Planning with stochastic program using estimated average of delays without predictions (i.e. traditional stochastic method), (2) Planning deterministically using the class predictions instead of the probabilistic predictions, (3) Planning deterministically assuming a fixed delay of 1 day, which is the typical business practice, (4) Planning with perfect information on delays. We show that the proposed approach yields up to 2.07% improvement on total costs compared to the traditional stochastic approach, up to 2.11% compared to using class predictions, and up to 4.57% compared to the business practice. As expected, the proposed approach performs worse than planning with perfect information by an average of 1.06% and a maximum of 1.72%. We note that the proposed approach achieves 74% of the potential cost reduction from the business practice to the perfect information case.

Our paper has contributions both on the modeling and on the methodological side. On the methodological side, we propose using the probabilistic predictions of a decision tree in the scenario generation process of a 2-stage stochastic program. We separately demonstrate the benefits of using the predictions of a decision tree, and the benefits of using probabilistic predictions instead of class predictions. On the modeling side, we apply the proposed method to a capacitated barge planning problem with uncertain container delays, which, to our knowledge, has not been studied in literature. This leads to practical contributions since we test the approach with a numerical study based on real data. We show how the proposed approach outperforms the industrial practice, and that it is possible to decrease trucking significantly.

Against this background, the paper is organized as follows: In Section 2, we review the relevant literature. In Section 3, the details of the problem context are provided in terms of the periodic activities, network structure and order requirements. In Section 4, the proposed approach and the methodology to test the approach is explained. More specifically, the details on training the decision tree, the mathematical formulation of the 2-stage stochastic program, the role of predictions in scenario generation and the iterative process of planning and simulation are provided. In Section 5, the details of computational experiment and the findings are discussed, and in Section 6 the paper is concluded.

2. Literature review

Our work demonstrates how a specific integration of machine learning and stochastic optimization can improve decision making in a variant of barge planning, which to our knowledge has not yet been studied. We first review the most contextually relevant studies to our capacitated barge planning problem in order to position the problem in barge planning literature (for a general review, see SteadieSeifi et al., 2014). Secondly, we discuss studies that combine machine learning and optimization under uncertainty in order to position the methodological contribution. Here, we primarily focus on the domains that are relevant to barge planning. Third, we extend our discussion to other domains. For a more comprehensive literature review on stochastic optimization, Powell (2019) and Chen et al. (2018) can be consulted.

Our *problem context* is similar to Gumuskaya et al. (2020a) except that our paper includes barge capacities. Ignoring capacities enables Gumuskaya et al. (2020a) to adopt a relatively simpler 2-stage stochastic MIP that can be solved fast. This allows for an increased number of instances and a detailed sensitivity analysis of the problem parameters. Zweers et al. (2020) study operational transport planning of containers in the hinterland of the Port of Rotterdam. Their context is similar to ours contextually and methodologically. Contextually, their network consists of an inland terminal and multiple terminals in a deep sea port.

Methodologically, they also employ stochastic programming and Sample Average Approximation (SAA). There are significant differences though. Firstly, Zweers et al. (2020) consider a single period while we study a multi period problem with time dimension. Secondly, they consider uncertain number of containers that can be loaded/unloaded at a given call. We consider no such restriction, but container arrival times are individually uncertain. Rivera and Mes (2017) study making the rotation plans between a single inland terminal and the port with a multi horizon approach and dynamic orders received through time. They use MDP formulation and employ Approximate Dynamic Programming (ADP) to solve the problem, while we use an iterative 2-stage stochastic program in combination with a simulation. Fazi and Roodbergen (2018) also assume a network with one inland terminal and multiple terminals located in the port in a deterministic environment. They plan the round trips of multiple barges with fixed capacity, and formulate the problem as a bin packing problem. Their main purpose is to analyze the impact of different demurrage and detention policies on barge planning (i.e. the cost of keeping containers longer than the allowed duration within or outside the port).

We next review the papers that integrate machine learning with optimization in related domains. Bhattacharya et al. (2014) illustrate a conceptually similar example of combining optimization with machine learning to this study. Their methodology consists of two stages: first, they predict the traffic flow in the road network using support vector mechanisms. Second, they optimize the schedules of railway services with a mixed integer program taking the predictions of road traffic into account. Irannezhad et al. (2020) model the hinterland logistics of a port as a multi agent simulation model, where the agents act based on reinforcement learning. A central optimization algorithm provides decisions to the agents that minimize the total network costs. The agents, i.e. transport operators, may decide to sign up to join the central decisions or they may schedule their activities individually in a probabilistic behavior. van Riessen et al. (2016) propose using decision trees to allocate the containers to multimodal services in the hinterland of the Port of Rotterdam. They propose training decision trees on the optimal solutions found by solving a number of problems offline previously. The decision tree captures the specific patterns of these good solutions, and is used online by the transport planners to allocate transport orders to hinterland services deterministically. Hottung et al. (2020) solve container pre-marshaling problem using deep neural networks trained on existing solutions. The neural network is then integrated into a tree search algorithm, which results in less than 2% optimality gap in real sized instances. Some papers use machine learning to make predictions without integrating it with optimization. For example, Barbour et al. (2018) focus on improving predictions of the expected time of arrivals of freight trains using machine learning algorithms, namely support vector regression. Similar to our study, they use specific information on trains such as train length, tonnage in addition to the traffic data. They fit a model on a large historical dataset and show that improvements over 20% are achieved compared to benchmark methods. Balster et al. (2020) develop a method to predict the expected time of arrivals (ETA) of freight in an intermodal hinterland transport system using random forests.

When we widen the scope and look at the papers outside the hinterland transport domain, there is considerable effort in methodological research on how to integrate machine learning and optimization under uncertainty. Crespo-Vazquez et al. (2018) study maximizing the net income of power plants under uncertain available wind energy and price. Similar to us, they propose a method that relies on ML to detect the patterns in uncertain variables. More specifically, they use multi-variate clustering and recurrent neural networks to determine the probability of occurrence of scenarios. Huang et al. (2016) is relevant to our paper in the sense that it involves a 2-stage stochastic program, a decision tree, a Monte Carlo technique and simulation. However, in their paper and in some others (Şafak et al., 2018; Ridier et al., 2016; Hu and Hobbs, 2010), decision trees are employed to describe the structural representation of scenarios rather than as a prediction method. For example, Casey and Sen (2005) propose an algorithm to generate scenario trees for multi stage stochastic programs, which provides asymptotic convergence and a measure of optimality of the solution. Smirnov and Huchzermeier (2020) model workforce planning as a stochastic program and use regression trees to predict uncertain service times. They propose a heuristic to solve the stochastic program, in which the probabilities of the scenarios are characterized by the predictions. Mitra et al. (2019) show that it is possible to yield more representative scenarios using simple linear regression in the domain of performance management. Guevara et al. (2020) propose a method for making investment decisions in the energy sector using distributionally robust optimization (DRO) and Extreme Gradient Boosting (XGBoost), which is a predictive model based on a regression tree model. Here, the main role of ML is to determine the most significant uncertain variables that will be used to determine ambiguity sets. Bertsimas et al. (2018) and Tulabandhula and Rudin (2014) also propose data-driven methods to design uncertainty sets for robust optimization. Our study differs from these studies in the way in which we employ decision trees to make predictions *individually* for each uncertain variable, which are subsequently used to generate scenarios using the Monte Carlo technique. As such, instead of determining the probability of occurrence of each scenario, the scenarios are generated such that they incorporate the patterns in reality. By solving a two-stage stochastic program with these scenarios via SAA, we find solutions that are better suited to reality, and thus lead to less costs.

Against this background, our contribution in this paper is two-fold. First, on the methodological side, we propose a method that combines decision trees with stochastic programs to incorporate predictions for each uncertain variable into the stochastic program. Specifically, we train decision trees on historical data to predict container delays, and use the probabilistic predictions to create more accurate scenarios while solving a 2-stage stochastic program with SAA. Note that, we do not propose a specific scenario generation or reduction technique per se; for which, Mitra and Domenica (2010) and Löhndorf (2016) can be consulted for a literature review (also see Pflug and Pichler (2016) for a non-parametric scenario generation technique from empirical observations). Second, on the modeling side, we model and solve the capacitated barge planning problem with uncertain container delays, which, to the best of our knowledge, has not yet been studied in literature.

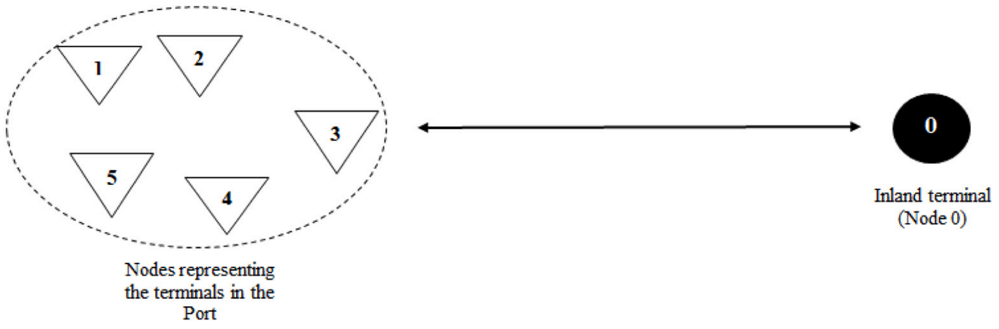


Fig. 1. An example representation of the network with 1 inland terminal and 5 terminals in the port (Gumuskaya et al., 2020a).

3. Problem setting

In this section, we provide the details of the problem context and the activities of the barge operator in a given period. This problem is based on an inland terminal that operates barges with a similar context to that of Gumuskaya et al. (2020a). In contrast to Gumuskaya et al. (2020a), in this paper we explicitly consider the barge capacities in the planning stage in order to make the problem setting more realistic and the solution approach applicable. While we formulate the problem in this domain, the proposed approach can be applied in other domains, where 2-stage stochastic programs are used and the uncertain variables can be predicted effectively (by decision trees).

We assume a network N, consisting of terminals $k \in N$ as nodes, and a single barge with capacity Q making trips between a single inland terminal ($k = 0$) and terminals located in the port ($k \in \{1, 2, \dots, |N| - 1\}$) (see Fig. 1). One round trip takes two days, and a port visit is immediately followed by an inland terminal visit on the next day or vice versa. Thus, a container will be delivered to its destination 1 day after its actual pick up date from its origin. For example, one round trip is as follows: The first day, the barge arrives at the port and spends 24 h in the port for the calls. The second day it sails to the inland terminal, loads/unloads the containers and sails back to the port, which in total takes another 24 h and then the pattern repeats itself. The maximum number of calls in a port visit is limited due to factors such as sailing time within the port, waiting times between the calls, disturbances in call handlings or disruptions (Douma et al., 2009; Zweers et al., 2020; Gumuskaya et al., 2020a).

The barge operator decides which terminals (k) to make calls (i.e. visits) at which days ($t \in \Psi_p$) in a given period p (i.e. makes the barge plan), represented by the decision variable X_k^t. In the course of events, terminals oblige the barge operator to finalize the barge plan a few days in advance. We assume this decision making moment to be discrete at the start of each period, which is a common assumption in literature (van Riessen et al., 2016; Lium et al., 2009; Fazi et al., 2015). Each call incurs a cost of C^B that represents the fuel cost of the additional travel from one terminal to another. Note that typically the fees for loading/unloading the containers are charged to the customer by the container terminals, and thus not included as a cost item for the barge operator.

At the start of each period p, the barge operator has a list of transport orders (I_p). These orders can be either *import* orders (i.e. from the port to the inland terminal) or *export* orders (i.e. from the inland terminal to the port). Each order i has an *expected* arrival date a_i at its origin node $o_i \in N$, and needs to be delivered to its destination node $d_i \in N$ with a strict due date b_i. As discussed in Section 1, due to the delays in deep sea vessel arrivals or preparation of cargo, the arrival dates of containers are subject to uncertain delays. Due dates, on the other hand, are strict. In the cases where the due dates are not met, the container has to be trucked immediately, and each truck incurs a cost of C^T. Trucking a container may be intentional due to the barge operator's plan or it may be caused by a delay in container's arrival. If a container i has a due date beyond the planning horizon, there is no need for trucking immediately as it may be postponed to the next period in the hope that it will be picked up by barge then. Gumuskaya et al. (2020a) show that when containers are freely postponed, an excess number of containers are left for next period, many of which need to be trucked eventually. They illustrate that introducing an artificial penalty cost per container avoids myopic solutions. Therefore, in this paper, we also include a penalty cost of C^P per container.

Next we discuss the overall methodology, the details of predicting delays by the decision tree, 2-stage stochastic program that is used for planning, and simulation used to determine actual events and evaluate costs.

4. Methodology

The main idea in this paper is to use the predictions of a decision tree in the scenario generation process of a stochastic program. These predictions help to generate more accurate scenarios, which in turn leads to a more informed decision and less cost.

The solution approach is modular in the sense that instead of using decision tree, any ML algorithm could be used as long as it can make probabilistic predictions. We selected decision trees in this case for two reasons. First, it is a well-studied and proven technique. For example, Dogan and Tanrikulu (2013) show that CART (Classification and Regression Trees) is the most robust among many machine learning methods. Likewise, Zhang et al. (2017) show that Stochastic Gradient Boosting Trees had the best accuracy among a selection of algorithms. The second reason is that our dataset includes both numerical and categorical data, which can be both handled well by decision trees. However, depending on the specific problem and prediction requirements, other ML methods can also be used, such as neural networks or Bayesian models.

Apart from the decision tree that predicts container delays, simulation is used. There are a few reasons why the objective function value of the stochastic program is not sufficient and a simulation stage is needed: first, for the purposes of this paper we need to check the long term performance. Obviously, a planning horizon of for example one year is not possible because the transport orders are not known such in advance. On the other hand, a multi stage stochastic program of this size would lead to tractability issues. Therefore a one-year run is made consisting of iterations with shorter planning horizons (i.e. 4 days). The continuum of events between the iterations are preserved by the simulation stage. For example, the containers that are postponed to the next period are determined by simulation, and those containers will be carried to next period. Second, the objective function of the stochastic program involves the artificial costs of postponing containers, meant to avoid postponing too many containers. These artificial costs do not exist in real life and the chosen value has a significant effect on the objective function. The real costs calculated in simulation, i.e. barge and truck costs, are a better indicator of performance. Third, the decision tree is meant to capture the delay patterns in real life and provide useful information to create more accurate scenarios. By using random samples from historical data, we aim to mimic real life with simulation and test if the decision tree would indeed capture such patterns.

4.1. Outline

Our solution approach mainly involves iteration of periodic activities (planning and simulating barge and truck trips) for a total duration of one year in order to observe the long term performance of the system (see Fig. 2). Prior to iterations, a decision tree (DT) is trained that predicts the probability of container delays based on container information (see Section 4.2 for details). This is done only once at the start of a run, and the same DT is used for predictions in the rest of the run. Afterwards, iterations take place, each having two main stages: First, planning of barge calls; and second, simulation of actual container arrivals and container handlings.

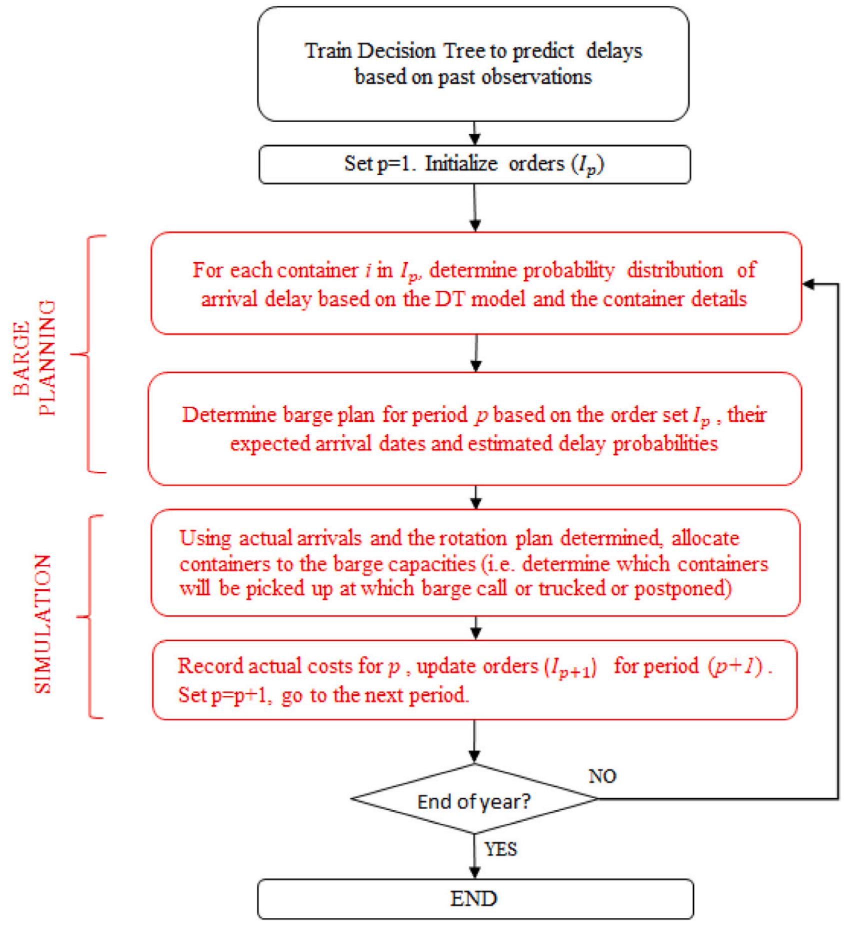

Train Decision Tree to predict delays based on past observations
Set p=1. Initialize orders (I_p)
BARGE PLANNING
For each container i in I_p, determine probability distribution of arrival delay based on the DT model and the container details
Determine barge plan for period p based on the order set I_p, their expected arrival dates and estimated delay probabilities
SIMULATION
Using actual arrivals and the rotation plan determined, allocate containers to the barge capacities (i.e. determine which containers will be picked up at which barge call or trucked or postponed)
Record actual costs for p, update orders (I_{p+1}) for period ($p+1$). Set p=p+1, go to the next period.
End of year?
NO
YES
END

Fig. 2. The representation of the steps of the overall solution procedure.
Source: Modified from Gumuskaya et al. (2020a).

At the start of each iteration p, there is a set of transport orders I_p, which include: those that are expected to arrive during period p (i.e. $a_i \in \Psi_p$) and the ones that were expected to arrive in previous periods (i.e. $a_i < min\{\Psi_p\}$) but postponed. The containers that were postponed may have already arrived at period $(p-1)$ (but not delivered to their destination), or they may be delayed and still expected to arrive some day in period p. For each container $i \in I_p$, the probability of delays is predicted using the DT. Note that rather than the class predictions (e.g. i will be delayed for 1 day), the probabilistic predictions are used (e.g. i will be delayed for 0 days with 5% probability, and for 1 day with 95%). This is an important aspect of the proposed approach: This way, reliable and unreliable predictions are implicitly distinguished and overreliance on inaccurate predictions is avoided in scenario generation. For example, assume that a container A is predicted to be delayed for 0 days with 5% and 1 day with 95%. Also assume another container B with class predictions (45%, 55%). Assuming that container B will be delayed by 1 day would lead to an inaccuracy of 45%, while for container A this is only 5%. Instead of using the single class prediction (i.e. 1 day for container A and 1 day for container B) deterministically, we use the probabilistic predictions to generate scenarios in order to hedge against such inaccuracies.

The barge plan is made taking into account the order set, I_p, and the probabilities of delays. These probabilities are used as the input for the scenario generation of a 2-stage stochastic MIP, which is solved to determine the barge plan (see Section 4.3 for details). The barge plan only dictates which terminals will be visited at which day; the allocation of containers to individual barge trips depend on the actual container delays. After the plan is made, actual events and costs are simulated for period p. During the simulation, actual delays from historical data are randomly used. Based on the barge plan and the actual delays, the allocation of containers to the barge capacities is made using a modified version of the 2-stage stochastic MIP (see Section 4.5). This model is actually meant to find what would the second stage variables be in the 2-stage stochastic program, given the barge plan and the actual arrivals. At this step, the actual handling of containers is determined (i.e. whether a container will be barged/postponed/trucked) and the actual costs are incurred. As a result of handling decisions, I_{p+1} is formed, which consists of new orders with expected arrival dates in period $p+1$ and the postponed orders from period p. The determination of the barge calls are the first stage variables, and the allocation of containers to the available capacities and determination of container handlings are the second stage variables of the stochastic program.

In the next sections, we will explain the details of training the decision tree, the mathematical formulation of the 2-stage stochastic MIP and the simulation.

4.2. Prediction of delays using a decision tree

In this section, we give details on the construction of the decision tree that predicts the container delays. The decision tree uses *the number of days a container is delayed* as the predicted class, referred to as "ACTUAL DELAY" for convenience. This design choice is straightforward because the time in both the stochastic program and the simulation is in days. In the computational experiment of this study, there are four classes that represent the delay: 0, 1, 2 and 3 days. The features that are used to make the predictions are "TERMINAL NAME", "REEFER INDICATOR" and "DEEP SEA VESSEL CALL SIZE", as summarized in Table 1. "TERMINAL NAME" is considered to be important since each container terminal has different operating strategies and service levels. "REEFER INDICATOR" refers to whether a container is refrigerated or not. It may be important because reefer containers are typically more expensive to store, and contain perishable items that need fast deliveries. Finally, "DEEP SEA VESSEL CALL SIZE" refers to the total number of containers that will be unloaded from a deep sea vessel. It may be important because the expected time to unload a container increases with the total number of containers the deep sea vessel needs to unload. When there are more containers, it takes more time to unload all containers since the container terminal has a limited capacity.

Table 1
The list of features used for training and testing of the decision tree and explanations.

Variable	Type	Explanation
ACTUAL DELAY	Categorical	The delay (in days) with respect to the reported expected arrival time.
TERMINAL	Categorical	The name of the terminal, to which the deep sea vessel drops the container.
REEFER	Binary	Represents if the container is a refrigerated container (e.g. for food requiring cold chain).
DEEP SEA VESSEL CALL SIZE	Integer	The total number of containers that are unloaded during the same deep sea vessel call.

The training set for the decision tree is separated from the dataset that is used for the computational experiments, in which the performance of the planning algorithm is evaluated (see Section 5.1). This is done because the planning algorithm uses predictions that are made by the decision tree. Using the same data for training the decision tree and for computational experiment would lead to an unfair advantage. For this reason, the training set excludes the containers used in the computational experiments. As a result, approximately 46% of the historical data is used in training set, consisting of around 23,000 rows.

Table 2
An illustration of the output of the scenario generation process using the decision tree.

Container ID	Terminal name	Reefer	Deep sea vessel call size	Delays (in days) for different scenarios									
				1	2	3	4	5	6	7	8	9	10
C1	Terminal 1	Yes	2500	2	3	2	2	1	3	2	2	2	3
C2	Terminal 1	Yes	2500	3	2	3	2	2	2	2	3	2	3
C3	Terminal 2	No	350	1	0	0	1	1	0	0	1	0	0

Table 3
Average execution times of one instance (i.e. a one-year run analysis) for the five methods in seconds.

Method	Computation time in seconds
DT-Stochastic	1648.3
Stochastic	1634.9
DT-Deterministic	36.0
1-Day Delay	62.2
Perfect Information	46.9

The decision tree is constructed using the "caret" package of R,[2] which uses the "rpart" library.[3] The splitting criterion used for training the decision tree is "Information Gain", the metric used is the "accuracy" determined by 10-fold cross validation, repeated three times. The pruning of the tree is done while the tree is constructed via a complexity parameter, cp, which checks if the added complexity (i.e. more splits) is justified by the improved misclassification error. The metric checked is the sum of two terms: misclassification error (referred to as cost) and the number of splits multiplied by the cp. For a prospective split to be accepted, the decrease in misclassification error (i.e. first term) should be higher than the increase in complexity (i.e. second term). If there is no such split, the tree growth is stopped. So, a higher cp makes it harder to add new splits and results in smaller trees. The details of the algorithm are excluded from this paper for brevity, and can be found in the documentation.[4] For a given training set, we try 10 different values for cp and pick the one that results in the best *accuracy*. The *accuracy* of a given tree is the overall agreement rate among three repetitions of 10-fold cross validation. The decision tree in this configuration led to an overall accuracy of around 66%. Considering that there are four classes to be predicted (0, 1, 2, 3), this is a satisfactory performance for the purposes of this paper.

4.3. *Mathematical model of the stochastic program*

The planning problem is to determine the calls of a capacitated barge for a given order set under uncertain delays for the specified planning horizon. The objective function is the minimization of expected cost of barge costs, trucking costs and penalty of postponing. In the mathematical model, the first stage decision variables (X_k^t) determine if terminal k will be visited at day $t \in \Psi_p$, where Ψ_p is the set of days to be planned for period p. The second stage decision variables represent container handlings for a given scenario ω: whether a container i will be picked up by barge at day t ($B_{i,t}^{\omega}$), whether it will be trucked (Y_i^{ω}), or postponed (Z_i^{ω}). Here a scenario ω represents a random realization of delay for each container (i.e. a given scenario consists of $|I_p|$ realizations in total). We use Sample Average Approximation (SAA) that uses the probabilistic predictions of the decision tree to solve the model. So we create a number of scenarios, each of which involves one realization for each random variable. Then, we optimize the mathematical model considering all of the generated scenarios. Scenario generation and the role of predictions are explained in Section 4.4.

[2] https://cran.r-project.org/web/packages/caret/caret.pdf.
[3] https://cran.r-project.org/web/packages/rpart/rpart.pdf.
[4] https://cran.r-project.org/web/packages/rpart/vignettes/longintro.pdf.

Notation

Sets

N	:	Set of nodes, $\{0\}$ for inland terminal, other terminals in the port $\in \{1, 2, \dots, \lvert N-1\rvert\}$
Ψ_p	:	Set of upcoming days to plan
ψ_{start}	:	The first day in Ψ_p, $\psi_{start} = min\{\Psi_p\}$
ψ_{end}	:	The last day in Ψ_p, $\psi_{end} = max\{\Psi_p\}$
I_p	:	Set of all orders for period p. $I_p = I_p^{I,P} \cup I_p^{I,T} \cup I_p^{E,P} \cup I_p^{E,T}$
$I_p^{I,P}$	:	Set of import orders that can be postponed to next period, i.e. $b_i > \psi_{end} + 1$.
$I_p^{I,T}$	:	Set of import orders that cannot be postponed to next period, i.e. $b_i \le \psi_{end} + 1$.
$I_p^{E,P}$	:	Set of export orders that can be postponed to next period, i.e. $b_i > \psi_{end} + 1$.
$I_p^{E,T}$	:	Set of export orders that cannot be postponed to next period, i.e. $b_i \le \psi_{end} + 1$
Ω	:	The set of generated scenarios, $\omega \in \{0, 1, 2, \dots, \lvert\Omega\rvert - 1\}$

Decision Variables

X_k^t	:	Decision variable representing barge calls, 1 if a call is made at node k at time t, $k \in N$, $t \in \Psi_p$
$B_{i,t}^{\omega}$	:	Decision variable representing the pick up time of a container by barge, 1 if container i is picked up at time t in scenario ω, $i \in I_p$, $\omega \in \Omega$, $t \in \Psi_p$
Y_i^{ω}	:	Decision variable representing trucking. 1 if container i is trucked in scenario ω, $i \in \{I_p^{I,T} \cup I_p^{E,T}\}, \omega \in \Omega$
Z_i^{ω}	:	Decision variable representing postponed orders. 1 if container i is postponed to next period in scenario ω, $i \in \{I_p^{I,P} \cup I_p^{E,P}\}, \omega \in \Omega$

Parameters

τ_i^{ω}	:	The number of days a container i is delayed in scenario ω, $\omega \in \Omega$
a_i	:	The expected arrival date of order i, $i \in I_p$
b_i	:	The due date of order i, $i \in I_p$
o_i	:	The origin node of order i, $o_i \in N$, $i \in I_p$
d_i	:	The destination node of order i, $d_i \in N$, $i \in I_p$
C^B	:	The marginal cost of a call at a terminal
C^T	:	The cost of trucking per container
C^P	:	The penalty for postponing a container
Q	:	The capacity of the barge
σ_k	:	Weight of a node, which can be interpreted as congestion in the port, used to limit the number of calls that can be made within 24 h, $0 < \sigma_k < 1$ for $k > 0$, $\sigma_k = 1$ for $k = 0$, $k \in N$
Δ	:	1 if the barge made a call in inland terminal just before the planning horizon starts (i.e. at time $\psi_{start} - 1$). Used to link barge's last position in previous period $p - 1$ with first position in current period p

$$Minimize \quad C^B \sum_{\forall t \in \Psi_p} \sum_{\forall k \in N} X_k^t + \frac{C^T}{\lvert\Omega\rvert} \sum_{\forall i \in \{I_p^{I,T} \cup I_p^{E,T}\}} \sum_{\forall \omega \in \Omega} Y_i^{\omega} + \frac{C^P}{\lvert\Omega\rvert} \sum_{\forall i \in \{I_p^{I,P} \cup I_p^{E,P}\}} \sum_{\forall \omega \in \Omega} Z_i^{\omega}$$

Subject to:

$$\sum_{t=max\{a_i+\tau_i^{\omega}, \psi_{start}\}}^{min\{b_i-1, \psi_{end}\}} B_{i,t}^{\omega} + Y_i^{\omega} = 1 \qquad \forall i \in \{I_p^{I,T} \cup I_p^{E,T}\}, \forall \omega \in \Omega \tag{1}$$

$$\sum_{t=max\{a_i+\tau_i^{\omega}, \psi_{start}\}}^{min\{b_i-1, \psi_{end}\}} B_{i,t}^{\omega} + Z_i^{\omega} = 1 \qquad \forall i \in \{I_p^{I,P} \cup I_p^{E,P}\}, \forall \omega \in \Omega \tag{2}$$

$$B_{i,t}^{\omega} \le X_{o_i}^t \qquad \forall i \in \{I_p^{I,T} \cup I_p^{I,P}\}, t \in \Psi_p, \forall \omega \in \Omega \tag{3}$$

$$B_{i,t}^{\omega} \le X_{d_i}^{t+1} \qquad \forall i \in \{I_p^{E,T} \cup I_p^{E,P}\}, t < \psi_{end}, \forall \omega \in \Omega \tag{4}$$

$$B_{i,\psi_{end}}^{\omega} = 0 \qquad \forall i \in \{I_p^{E,T} \cup I_p^{E,P}\}, \forall \omega \in \Omega \tag{5}$$

$$\sum_{\forall i \in I_p} B_{i,t}^{\omega} \le Q \qquad \forall t \in \Psi_p, \forall \omega \in \Omega \tag{6}$$

$$\sum_{\forall k \in N} \sigma_k X_k^t \le 1 \qquad \forall t \in \Psi_p \tag{7}$$

$$X_0^t + X_0^{t-1} = 1 \qquad \forall t \in \Psi_p, t > \psi_{start} \tag{8}$$

$$X_0^{\psi_{start}} = 1 - \Delta \tag{9}$$

$$X_k^t, B_{i,t}^{\omega}, Y_i^{\omega}, Z_i^{\omega} \in \{0, 1\} \qquad \forall k \in N, \forall t \in \Psi_p, \forall i \in I_p, \omega \in \Omega \tag{10}$$

The objective function minimizes the expected sum of barge costs, trucking costs and penalty costs over all scenarios. The barge costs represent the marginal cost of making an additional call, and only depend on the decision of the operator. So it is not affected by the scenarios and do not need the index ω. Trucking costs are direct costs of hiring an external truck operator, while penalty costs are introduced for postponed containers to avoid myopic solutions. Both trucking costs and penalty costs depend on the delay scenario. Therefore, associated variables Y_i^ω and Z_i^ω are defined for each container and scenario. These are then multiplied by the unit costs (C^T and C^P) and the probability that a given scenario occurs, $1/|\Omega|$, to find the expected costs. Note that the number of trips between the inland terminal and the port is fixed due to the alternating schedule of the barge, and thus associated sailing costs (i.e. the cost of long hauls between the port and the inland terminal) are excluded. The optimization is subject to the following constraints.

(1) Each container i that is not allowed to be postponed (i.e. $b_i \leq \psi_{end} + 1$) must be picked up by a barge earliest when the container is ready (i.e. at τ_i^ω days after its expected arrival a_i) or trucked for a given scenario ω. The barge pick up has to be within the given period and latest one day before the due date (sailing from the inland terminal to the port takes 1 day). Note that the containers with a due date of $\psi_{end} + 1$ also cannot be postponed; even if the container is picked up at the first day of the new period, it will take 1 day to sail and the container will be delivered in the second day, which violates the due date.

(2) Similar logic with constraint set (1) applies. Each container i that is allowed to be postponed (i.e. $b_i > \psi_{end} + 1$) must be picked up by a barge earliest when the container is ready (i.e. at τ_i^ω days after its expected arrival a_i) or postponed to next period for a given scenario ω. The barge pick up has to be within the given period and latest one day before the due date (sailing from the port to the inland terminal takes 1 day).

(3) Logical constraints to make sure that if an import container i is picked up by a barge at time t, there has to be a barge call at time t at the origin terminal o_i.

(4) Logical constraints to make sure that if an export container i is picked up by a barge at time t (to be delivered at time $t+1$ to node d_i), there has to be a barge call at time $t+1$ at the destination terminal d_i. Note that when an export container is picked up at the last day of the planning horizon (at day ψ_{end}), we do not know if the barge will make a call in the destination terminal at the first day of the next period (at day $\psi_{end} + 1$), so we do not allow pick up. This is enforced by constraint set (5).

(6) The total number of containers picked up at a given time cannot exceed the barge capacity. Those containers will sail either from the port to the inland terminal or vice versa in the following day.

(7) The total duration in the port cannot be exceeded. σ_k values can be interpreted as the congestion in the port. With an increased value of σ_k, the number of calls that can be made within 24 h decreases.

(8) The barge has to alternate between the port and the inland terminal in consecutive days.

(9) Linking of the barge call in the first day of the planning horizon with the last day of the previous period.

(10) Binary variables for barge calls, barge pick ups, trucking variables and postponing variables.

4.4. Scenario generation with predictions

In this section, we explain why SAA is used for solving the stochastic program, provide details on scenario generation process and the role of decision trees in it.

The number of decision variables in the stochastic model increases with the number of scenarios, which causes longer computational times. For example, assuming that there are d possible delays for a given container (i.e. $0, 1, 2, \ldots, (d-1)$ days of delay) and $|I_p|$ number of orders, the number of possible scenarios would be $d^{|I_p|}$, which grows exponentially with the number of orders and leads to intractable computational times. To overcome this issue, SAA is frequently used in stochastic programs in literature (Shapiro et al., 2014). In a nutshell, instead of solving the problem with the full set of scenarios, a much smaller number of scenarios is generated and the resulting model is solved.

In this paper, we also employ SAA and generate scenarios by using Monte Carlo technique (see Mak et al., 1999). The scenario generation uses the predictions of the decision tree as follows: For a given period, a single scenario includes $|I_p|$ generated delays in total (i.e. one delay generated per order). Similar to Gumuskaya et al. (2020a), we assume the delays are random variables following a discrete probability mass function (pmf). The decision tree is used at this stage to determine pmf of individual delays in order to generate random realizations for constructing scenarios. For each order, the decision tree provides the probability that the container will be delayed by d days, and these probabilities are used to generate random realizations for each order. A scenario is completed when one random realization is generated for each order. A sample output of the scenario generation is illustrated in Table 2, where there are 3 containers, the number of scenarios is 10 and the delays are allowed to take values from the set $[0, 3]$ days. In the table, we see that the first two containers (i.e. "C1" and "C2") have the same features and thus follow the same pmf, which happens to generate long delays. On the other hand, the delays of the last container (i.e. "C3") is generated from a completely different pmf, which generates shorter delays.

Note that while generating a random realization for a given container and its given pmf, some of the delay scenarios may not be feasible. For example, a container that was expected to arrive in the previous period may already be delayed for a few days. Another example could be that the container has already arrived, so there is no reason to expect new delay. In such cases, conditional probabilities are used to generate scenarios as in Gumuskaya et al. (2020a).

4.5. Simulation of actual events

Up until now, the decision tree is trained on historical data and the barge plan is made by the stochastic program. As explained at the beginning of Section 4, simulation of actual events is needed to test the long term performance, to avoid the impact of artificial costs and to mimic real life patterns by random sampling the delays from historical data. These events are as follows: containers arrive with or without delays, the barge calls are made, containers are picked up during the barge calls, certain containers are trucked and certain are postponed to next period. We provide the details of simulation in this section.

The container delays are determined by random sampling from historical data as explained in Section 5.1. Since expected arrival dates are known, the actual arrival is simply found by adding the delay to the expected arrival. The barge call decisions, which correspond to the first stage decision variables in the stochastic program, are conclusively made in planning stage and cannot change. So independent of other events, barge calls will be made according to the plan. Finally, whether a container will be picked up by a barge or truck or postponed corresponds to the second stage variables, and depends on the delay scenario. However, the actual delays sampled from the historical data most likely form a new scenario that is not one of the scenarios generated in the stochastic program. To determine what would happen in the actual scenario, we solve the stochastic program by fixing the first stage decision variables (i.e. X_k^t) to the values already determined in the planning stage. Hence, the same stochastic model will be solved considering one single scenario (i.e. the actual scenario) as provided below. Formally, the mathematical model in Section 4.3 is modified as follows:

$$Minimize \quad C^B \sum_{t\in\Psi_p} \sum_{k\in N} X_k^t + C^T \sum_{\forall i} Y_i + C^P \sum_{\forall i} Z_i$$

Subject to:

$$Constraints\ (1), (2), (3), (4), (5), (6), (10)$$

$$X_k^t = X_k^{t*} \qquad \forall k \in N, \forall t \in \Psi_p \tag{11}$$

where X_k^{t*} is the optimal solution vector found in the planning stage and all other decision variables are the same as described in Section 4.3 without the scenario index ω. Moreover, the container delays (τ_i) are set to the actual delays. Note also that since there is only one scenario, the expectation of trucking and postponing costs are not necessary (i.e. $1/|\Omega|$ is omitted).

By solving the above model, the container handling decisions (i.e. second stage decision variables) are made and the *Periodic Actual Cost* is calculated. The periodic actual cost consists of barge costs and the trucking costs, which is the sum of the first two terms in the objective function. The penalty of postponing containers is excluded because these are artificial costs. The postponed containers will eventually be barged or trucked in the following periods, which will then be accounted for as part of the corresponding periodic actual costs. Finally, the periodic actual costs are summed for all iterations to calculate the *Total Actual Cost* for a complete run of one year. By using this method of cost calculation by simulation, it is possible to isolate the impact of using the predictions of decision tree versus the traditional stochastic programming. As will be seen in Section 5, this way we can make a fair comparison using the same actual delays for all benchmark methods compared.

5. Computational experiment and results

In this section, we provide the details of the computational experiment that we use to test the performance of the proposed approach. In Section 5.1, first we explain how the dataset is formed, how parameters are set, and which methods are used for benchmarking. In Section 5.2, we discuss the estimated optimality gap of the stochastic program and the reliability of the scenario generation process. In Section 5.3, we compare the methods numerically and discuss our findings.

5.1. Setup

The computational experiment for this study necessitates a dataset consisting of orders with the following details: order requirements for planning (i.e. origin, destination, expected arrival, due date), container specific information for predictions (terminal name, reefer indicator, deep sea vessel call size) and actual delays for simulation. Such a dataset could be fabricated from scratch but this would hinder the substantiation of the study: the demand patterns (e.g. distribution of demand volume between origin–destination pairs) and the delay patterns would be up to user's choice. The decision tree would discover the patterns that are artificially introduced by the user. Using a complete real dataset is ideal, but we are not aware of the existence of such a complete dataset that is accessible for research. For this reason, we merge the real data of an inland terminal (Gumuskaya et al., 2020a) and the historical data of the Port of Rotterdam (Smulders, 2019). In a nutshell, the orders (i.e. origin, destination, expected arrival, due date) are taken directly from Gumuskaya et al. (2020a), which consists of 42,355 orders in total and there are 20 terminals. For each order, container specific information and actual delays are sampled from the historical dataset used in Smulders (2019). The containers, whose actual delays are used during this sampling are excluded from the training set of the decision tree to avoid unfair advantage. For brevity, we provide the details of the complete procedure in Appendix. Since the procedure involves random sampling (e.g. actual delays used in simulation stage), we create 10 different datasets to check if the findings will hold under different random samples (i.e. different datasets).

During the computational experiment, for each dataset five benchmark methods are applied and their performances are compared:

1. *DT-Stochastic*: The main objective of the computational experiment is to check how this method (i.e. using the predictions in the 2-stage stochastic MIP) performs with respect to other methods. This is the proposed approach as explained in Section 4.
2. *Stochastic*: This method represents the traditional stochastic approach without using the decision tree predictions during scenario generation. It is identical to *DT-Stochastic* except that in *Stochastic*, a fixed probability mass function is used for all orders, instead of the specific ones predicted by the DT. The probability mass function that is used for all the orders during generating scenarios is set to the overall percentage of delays in the historical data as follows:

$$P(D=d)=\begin{cases}8.1\%, & d=0\\ 41.1\%, & d=1\\ 35.1\%, & d=2\\ 15.7\%, & d=3\end{cases}$$

   Hence, by comparing *DT-Stochastic* with *Stochastic*, which are both stochastic approaches, we isolate the impact of using predictions in scenario generation.
3. *DT-Deterministic*: This method corresponds to using the class predictions in a deterministic way, instead of using the probabilistic predictions. The application of the method is as follows: the same mathematical model in Section 4.3 is constructed considering only one scenario, where all delays are equal to the predicted values by the decision tree. By comparing *DT-Deterministic* with *DT-Stochastic* we show the benefit of using stochastic approach for the same decision tree model.
4. *1-Day Delay*: This method is based on the industrial practice (Smulders, 2019). Barge operators usually assume that the containers will be released in 1 day, and make deterministic plans accordingly. The application of the method is similar to *DT-Deterministic*: the same mathematical model in Section 4.3 is constructed considering only one scenario, where all delays are equal to 1. By comparing *1-day delay* with other methods, we show how industrial methods perform.
5. *Perfect Information*: This method represents the hypothetical case where all delays are known with certainty before planning. So, it shows the best possible solution under a given set of actual delay scenario. It is solved similar to *DT-Deterministic* and *1-Day Delay* using the actual delays.

The parameters that characterize the instances are set using the findings of Gumuskaya et al. (2020a), where an extensive sensitivity analysis is made. Truck cost (C^T) is 250, and the penalty for postponing containers (C^P) is 100 since it proved to offer the minimum costs in many instances. Barge cost (C^B) is proven to be insignificant and set to 50 arbitrarily. The maximum number of calls was shown to be significant in the range [3,7]. In this paper, to reduce computational times, we set maximum number of calls to either 3 or 7. A given dataset is solved for each of the five methods (*DT-Stochastic, Stochastic, DT-Deterministic, 1-Day Delay and Perfect Information*). In conclusion, there are 10 datasets and 2 maximum number of call values (i.e. 3 or 7), leading to 20 instances solved for five approaches.

The solution approach is coded in C and the 2-stage stochastic MIP and the MIP in simulation stage were solved by invoking Gurobi 8.1 from C. The runs are executed on an Intel® Core™ i7 - 6700HQ with 2.60 GHz processor and 8 GB of RAM. The training of decision trees, cleaning of data and the preparation of dataset are executed in R 3.6.1 before the actual runs. The execution times of the instances in C are summarized in Table 3 for the five methods. Note that execution of *DT-Stochastic* and *Stochastic* took longer than *DT-Deterministic*, *1-Day Delay* and *Perfect Information* because in the former three, the model is much bigger due to the scenarios. Also note that execution times of one instance is the sum of solving for multiple iterations for the whole year.

5.2. Optimality gap estimate of the stochastic program

The determination of the number of scenarios used in stochastic program is done based on preliminary runs. In these runs, we test 6 different values from the set $\{2,5,10,20,50,100\}$. For each of them, we solve the stochastic program of an arbitrary period 30 times, and report solution times and the objective function values. Based on the trade-off, we picked 20 (see Appendix A.3 for details). We further estimate the optimality gap numerically to show that the solutions are reliable using the independent random streams method by Mak et al. (1999) (For an alternative method, see Kaut and Wallace, 2007).

Two runs are made to determine lower and upper limits for a given period p. In the first run, 2-stage stochastic model formulated in Section 4.3 is solved 25 times ($n_l = 25$) with different random number streams, and the mean objective function value is calculated as $\bar{L}_{(n_l)}$ to be used for lower limit calculation. In the second run, an optimal first stage solution $\hat{x}$ for a given 2-stage stochastic model is fixed. This solution is tested on a new independent generated scenario for 1000 times ($n_u = 1000$) to check the expected performance in long run. The mean value of objective function values is recorded as $\bar{U}_{(n_u)}$ and used to determine upper limit. The optimality gap is formulated by the following formula:

$$\left[0, \left(\bar{U}_{(n_u)} - \bar{L}_{(n_l)} + \varepsilon_u + \varepsilon_l\right)\right]$$

$$\varepsilon_l = \frac{t_{n_l-1,\alpha} s_l(n_l)}{\sqrt{n_l}}$$

$$\varepsilon_u = \frac{t_{n_u-1,\alpha} s_u(n_u)}{\sqrt{n_u}}$$

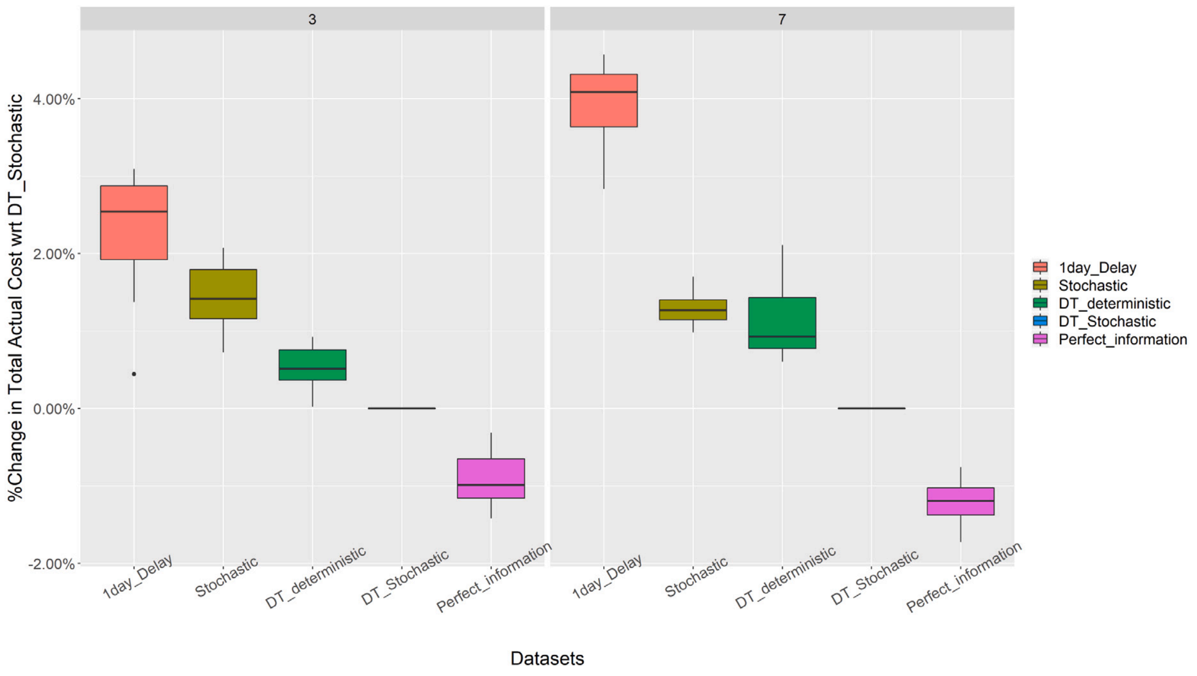


Fig. 3. Box plots of pairwise comparison in total actual costs wrt *DT-Stochastic* for different maximum number of calls per visit.

where $s_l(n_l)$ and $s_u(n_u)$ are the standard deviations of the objective functions values, t is the student t-distribution and α is the confidence level. Recall that this complete procedure is applied for a single stochastic program and thus, a given period p. To check if the optimality gap does not depend on the chosen period, we made the above test for three randomly chosen periods. For these three periods, with the number of scenarios equal to 20 and confidence level of 95%, the optimality gap turned out to be 0.07%, 0.12% and 0.27%, which shows that the scenario generation process is reliable with 20 scenarios. Therefore, we use 20 scenarios for each stochastic program in the computational experiment.

5.3. Results

5.3.1. Performance of the proposed approach

The evaluation of the methods is performed based on *total actual costs* and *net total actual costs*. *Total actual costs* is the total sum of all trucking and barge costs over all periods. However, using only total costs may be misleading because it includes trucking costs that are inevitable irrespective of the chosen method: when the delay of a container is so high that the due date is already violated when the container is ready to be picked up, it is impossible to use barge. Hence, whatever barge plan is made or whether the delay is predicted perfectly, these containers will be trucked. Since our objective is to compare the effectiveness of various methods, these costs are excluded from *total actual costs* and reported as *net total actual costs*.

In Fig. 3, the pairwise comparisons in total actual costs with respect to *DT-Stochastic* (the proposed approach) are plotted. Each box plot involves the difference between the costs of a given method and *DT-Stochastic*, divided by the cost of *DT-Stochastic* shown in percentage (So a positive value implies *DT-Stochastic* has a lower cost). Since there are ten datasets, each box is drawn using 10 pairwise comparisons found accordingly. The two maximum number of call values are separated in two plots side by side. From the figure, we observe that the main idea of this paper is confirmed as *DT-Stochastic* performs significantly better than all the other methods except *Perfect Information*. *Perfect Information* yields the best costs, which is expected. We also observe that the business practice, i.e. *1-Day Delay*, performs the worst. This is due to the fact that it does not take into account the stochasticity at all nor does it use predictions.

Fig. 3 gives an idea of the differences between the costs of the methods but does not indicate the amount of the exact improvements. For this reason, in Table 4, the pairwise comparisons with respect to *DT-Stochastic* are summarized numerically. Overall, the proposed approach leads to an improvement in total actual costs by a maximum of 2.07% and 1.37% on average compared to *Stochastic*. When we look at the net total actual costs, the improvements are maximum 4.18% and 2.79% on average. This shows that using the predictions with a stochastic approach indeed helped to decrease the costs. On the other hand, when compared with *DT-Deterministic*, the proposed approach decreases the total costs by a maximum of 2.11% and 0.83% on average. In net costs these numbers are 5.17% and 1.83% respectively, which substantiates using the probabilistic predictions instead of class predictions. Business practice (i.e. *1-Day Delay*) performs significantly worse than *DT-Stochastic* by 3.09% in total actual costs and 6.65% in net total actual costs on average. As expected, *Perfect Information* outperforms all of the methods and illustrate the

Table 4
Summary of percent changes in costs of the methods w.r.t *DT-Stochastic* method. Positive values imply an increase in costs, and thus worse performance w.r.t *DT-Stochastic*.

Method	Total cost				Net Total Cost			
	Average value	%Change w.r.t DT-Stochastic			Average value	%Change w.r.t DT-Stochastic		
		Max	Min	Average		Max	Min	Average
1-Day Delay	4386910	4.57	0.44	3.09	2343485	11.19	0.74	6.65
Stochastic	4320613	2.07	0.72	1.37	2277188	4.18	1.21	2.79
DT-Deterministic	4294770	2.11	0.02	0.83	2251345	5.17	0.03	1.83
Perfect information	4217770	−0.31	−1.72	−1.06	2174345	−0.53	−4.26	−2.24
DT Stochastic	4261665	0	0	0	2218240	0	0	0

potential to improve. *Perfect Information* improves *DT-Stochastic* by an average of 1.06% in total actual costs and 2.24% in net total actual costs.

These numbers clearly demonstrate the benefits of using probabilistic predictions with a stochastic perspective in hinterland transport domain, which is the main purpose of the computational study. If we look at the room for improvement between the worst (i.e. *1-Day Delay*) and the best method (i.e. *Perfect information*), we see that the proposed approach realizes roughly 74% of this margin. Therefore, the numbers in Table 4 would inflate further when the improvement potential is higher.

5.3.2. The relation between the accuracy of the decision tree and the performance

The performance of the overall approach depends on the ability of the decision tree model to capture patterns in data and create accurate predictions. In this section, we present more experiments to analyze the relation between the performance of the overall solution approach and the accuracy of the decision tree model.

In these additional experiments, we created a new original dataset in the same manner explained in Section 5.1, and trained a decision tree model (DT). The accuracy of the algorithm turned out to be 62.5%. Afterwards, to simulate varying levels of accuracy, we manipulated the predictions in the dataset and reported the performance of each level using the following procedure:

- A set of discrete numbers is created from 100 to 0 with increments of 10 (i.e. $\{100, 90, 80, \dots, 10, 0\}$). Each number represents the percentage of orders that use the (more accurate) DT predictions.
- For the remaining orders, the (less accurate) default probabilities of the traditional stochastic approach are used (i.e. $\{8.1\%, 41.1\%, 35.1\%, 15.7\%\}$ for $\{0, 1, 2, 3\}$ days of delay). So 100 refers to the case in which all orders use the probabilistic predictions made by DT model. In "Case 90", 90% of the orders use the predictions suggested by the DT model and 10% use the ones made by the stochastic approach. In our dataset there are approximately 42,000 orders in total over a year. For "Case 90", 4200 orders will use the default probabilities for generating scenarios of the two-stage stochastic program. The remaining 37,800 orders will use the probabilistic predictions made by DT model.
- For each case, the orders that use DT predictions and the orders that use default probabilities are selected randomly. However, the cases build on each other to eliminate random effects from one case to the next. For example, for "Case 90", 4200 orders are selected to use default probabilities. For "Case 80", out of the 37,800 orders that use predictions in "Case 90", an additional 4200 orders are selected to use default probabilities and so on.
- For each case, a complete run is made 10 different times using different random seeds, and the results are reported.

The findings are illustrated by Fig. 4 and Table 5. In Table 5, we also report the observed accuracy of the predictions. The overall prediction accuracy depends on the percentage of orders using decision tree as shown in Table 5.

We observe that although there are fluctuations due to randomness, the total costs decrease as the decision tree is used more, which visually seems to be a linear relationship. Interestingly, we see that even if the predictions made by the decision trees are poor, they still have benefits in terms of lower total costs. We also observe that "Case 0" has the same performance as pure stochastic approach. This is expected because in that case all orders use the default probabilities.

6. Conclusion

In this paper, we propose a solution approach that combines machine learning with combinatorial optimization. The approach involves making predictions for the uncertain variables of a 2-stage stochastic program using a decision tree, and applying these predictions to create more accurate scenarios. Instead of using the class predictions of the decision tree deterministically, the probabilistic predictions are used during the scenario generation, which allows hedging against the risk of inaccurate predictions. The proposed approach is tested on a capacitated barge planning problem with uncertain container arrivals. The methodology starts with training a decision tree and afterwards involves an iterative method, in which each iteration consists of a planning stage and a simulation stage. The planning stage corresponds to determining barge plan by solving a 2-stage stochastic program with Sample Average Approximation (SAA). Based on the solution of the stochastic program, actual events are simulated and costs are reported. Planning and simulation stages are repeated for a one-year duration in order to check the performance of the approach in the long run.

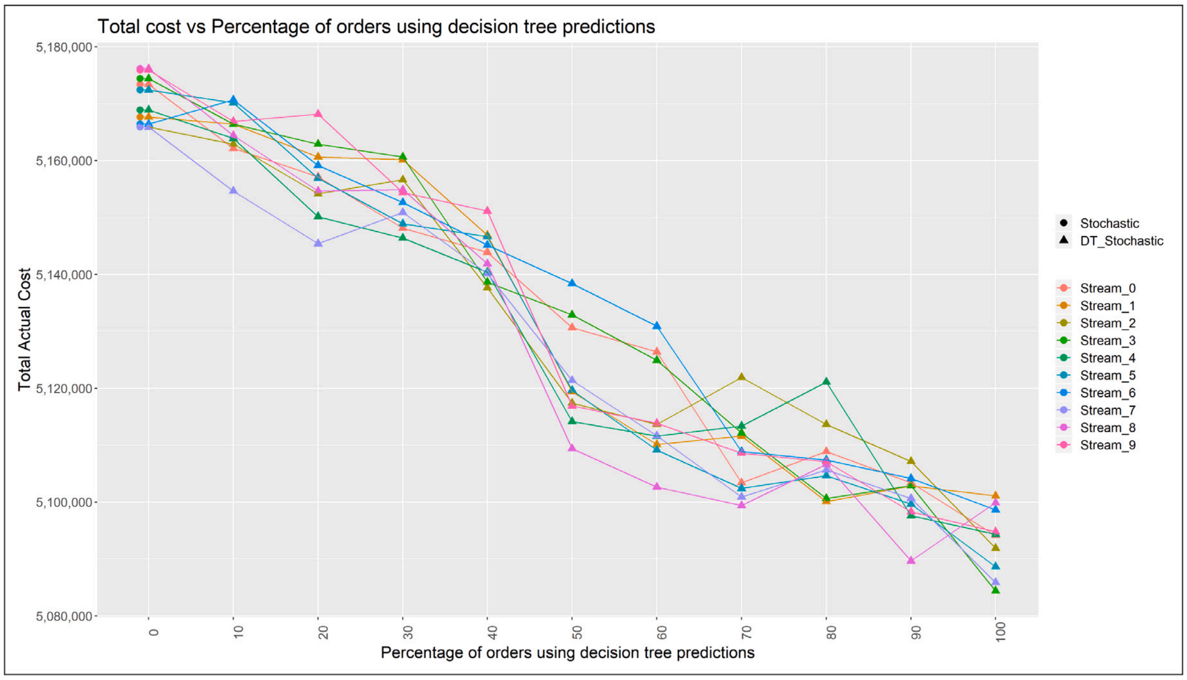


Fig. 4. The total actual costs per each case and random number stream.

Table 5
The total costs, net total costs and accuracy of the predictions for each case and method among 10 random number streams.

Method	Average total cost among 10 random number streams	Average net total cost among 10 random number streams	Prediction accuracy
1-Day delay	5220900	3154150	
Stochastic	5170700	3103950	
Case 0	5170700	3103950	40
Case 10	5164850	3098100	42
Case 20	5156925	3090175	44
Case 30	5153375	3086625	47
Case 40	5143250	3076500	48
Case 50	5122025	3055275	51
Case 60	5115485	3048735	53
Case 70	5108260	3041510	55
Case 80	5107585	3040835	57
Case 90	5100630	3033880	60
Case 100	5093380	3026630	62
Perfect Information	5054800	2988050	

Using the historical data of an inland terminal and the Port of Rotterdam, a computational experiment is made. The results show that the proposed approach offers an average decrease of 1.37% in total costs and 2.79% in net total costs over traditional stochastic programming. We also show that using probabilistic predictions instead of class predictions, an average improvement of 0.83% in total costs and 1.83% in net total costs can be achieved.

These results have important implications for practice: first, the evaluation in this paper shows that explicitly considering uncertainty, even with a traditional stochastic program, leads to significant improvements over the current method, in which the planners assume deterministically that the containers will be ready 1 day after their expected arrival dates. This underlines the importance of a stochastic approach. Second, this research demonstrates a popular recent idea on freight transport domain: that (big) data can be used by companies to improve decision making through more accurate predictions of uncertain factors.

In principle our proposed method is applicable in many fields where two-stage stochastic programming can be used. The applicability of the method depends on whether the uncertain variables can be predicted with accuracy and whether the perfect information offers significant improvement. As long as these two conditions are satisfied, the scenarios would be more accurate leading to significant improvements. For example, in vehicle routing problems with uncertain travel times one may check for special characteristics of the travel times between the node pairs that can be predicted. If there is a football stadium or market creating congestion on specific days, this information could be used to make better predictions. Possibilities also exist for problems such as portfolio optimization or inventory control.

In this paper, the improvement potential by perfect information was roughly 4% on average, and the prediction accuracy of the decision tree was 66%. We believe the effectiveness of the proposed approach and its benefits could be more substantial in different domains, where the improvement potential by perfect information is higher. We made additional runs to check the relationship between the accuracy of the decision tree models and the overall performance. We observed that there seems to be a linear relationship between the accuracy and the overall performance, and that even poorer decision trees may lead to improvements.

A limitation of this paper is the usage of relatively simple decision trees. While we achieved effective results, other – more advanced – machine learning techniques, such as random forests, Bayesian models or neural networks can also be used. Potentially, such techniques can improve the prediction accuracy and optimization results further. However, the main purpose of this paper is not to create a state-of-the-art classification algorithm and the predictions were accurate enough to show significant improvements. An additional advantage of decision trees is that they are easy to understand and easy to implement by practitioners.

Another limitation of this work is that the data we had was limited. A more complete dataset, including other data sources such as the status of the deep sea vessel, weather conditions or information on the current status of the documentation process (e.g. customs or commercial release) could be more helpful. Future research can also be directed towards more complex transportation networks and stochastic demand.

CRediT authorship contribution statement

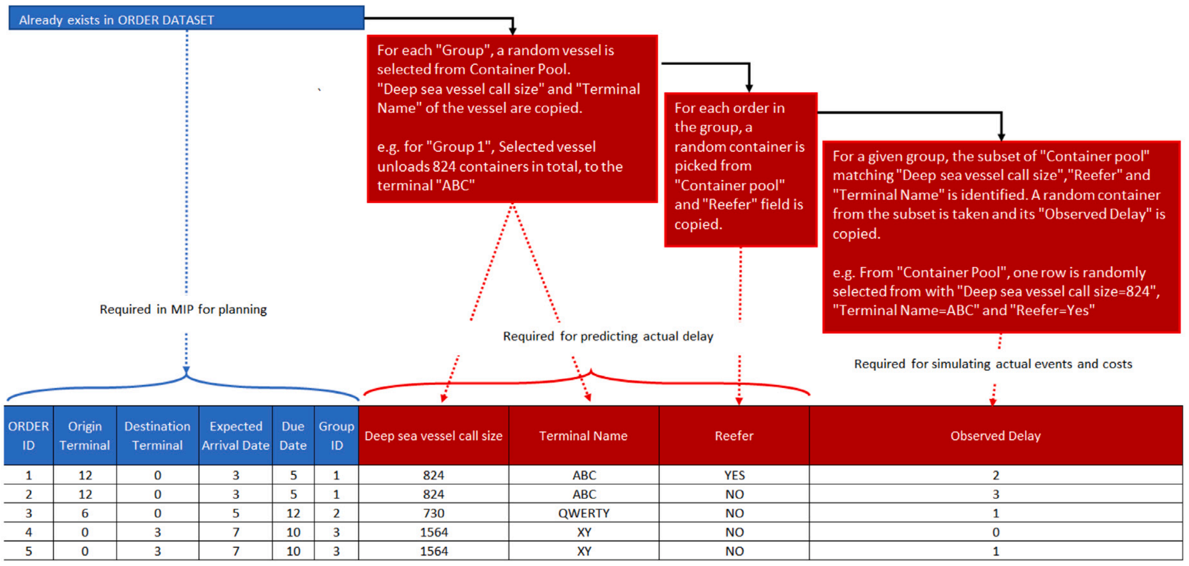
Volkan Gumuskaya: Conceptualization, Methodology, Software, Validation, Formal analysis, Resources, Writing – original draft, Visualization, Project administration. **Willem van Jaarsveld:** Conceptualization, Methodology, Investigation, Writing – review & editing, Supervision. **Remco Dijkman:** Conceptualization, Methodology, Investigation, Writing – review & editing, Supervision. **Paul Grefen:** Supervision, Writing – review & editing. **Albert Veenstra:** Supervision, Writing – review & editing.

Acknowledgment

This work is part of the ISOLA project with project number 438-13-214, which is partly financed by the Dutch Research Council (NWO), Netherlands.

Appendix

In this section, we provide the details of how datasets of the inland terminal and the Port of Rotterdam are merged in order to create the dataset used in computational experiments. While doing so, we make a serious effort to convert the raw data into usable datasets while being as loyal to the real data as possible. First, we introduce the original datasets. Then, we explain the shortcomings of these original datasets and the steps that are taken to convert them into a usable format for the computational experiment.


Already exists in ORDER DATASET
Required in MIP for planning
Required for predicting actual delay
Required for simulating actual events and costs

For each "Group", a random vessel is selected from Container Pool. "Deep sea vessel call size" and "Terminal Name" of the vessel are copied.

e.g. for "Group 1", Selected vessel unloads 824 containers in total, to the terminal "ABC"

For each order in the group, a random container is picked from "Container pool" and "Reefer" field is copied.

For a given group, the subset of "Container pool" matching "Deep sea vessel call size","Reefer" and "Terminal Name" is identified. A random container from the subset is taken and its "Observed Delay" is copied.

e.g. From "Container Pool", one row is randomly selected from with "Deep sea vessel call size=824", "Terminal Name=ABC" and "Reefer=Yes"

ORDER ID	Origin Terminal	Destination Terminal	Expected Arrival Date	Due Date	Group ID	Deep sea vessel call size	Terminal Name	Reefer	Observed Delay
1	12	0	3	5	1	824	ABC	YES	2
2	12	0	3	5	1	824	ABC	NO	3
3	6	0	5	12	2	730	QWERTY	NO	1
4	0	3	7	10	3	1564	XY	NO	0
5	0	3	7	10	3	1564	XY	NO	1

Fig. 5. The summary of data preparation steps.

Table 6
The conversion formula of the ACTUAL DELAY from hours to days.

x = Delay in hours	Delay in days	Percentage of observations in "CONTAINER_DATASET"
$x \leq 6$	0	8.1%
$6 < x \leq 30$	1	41.1%
$30 < x \leq 54$	2	35.1%
$x > 54$	3	15.7%

A.1. Introduction of the original datasets

There are originally three datasets: one from the inland terminal (ORDER_DATASET), and two from The Port of Rotterdam (CONTAINER_DATASET and DEEP_SEA_VESSEL_DATASET). The two datasets from the Port of Rotterdam are merged into a new dataset named CONTAINER_POOL, which is mainly used for training decision tree and prediction of delays. ORDER_DATASET consists of order requirements and acts as an input for the 2-stage stochastic MIP during the planning stage. The details of the datasets and data preparation steps are provided below:

1. CONTAINER_DATASET (source: Port of Rotterdam): Each row represents the most recent status of a single import container (e.g. arriving to the port) , and includes ***container specific*** data such as "Terminal", "Reefer", "Deep sea vessel call size", "Visit Call Reference Number", "Estimated Time of Arrival (ETA)", "Actual Time of Arrival (ATA)". If there are several notifications of a container made by the deep sea vessel such as updates of ETA due to estimated delays, the original data fields are overwritten and only the most recent ETA was kept. The further details of the dataset can be found in Smulders (2019), where it is originally collected and used.
2. DEEP_SEA_VESSEL_DATASET (source: Port of Rotterdam): Each row represents the status of a deep sea vessel at a given moment, and includes ***vessel specific*** data such as "Terminal", "Date of notification", "ETA", "ATA". If there are multiple notifications of the same deep sea vessel (e.g. ETA updates), new rows are inserted so that there are as many number of rows as the number of notifications. The further details of the dataset can be found in Smulders (2019), where it is originally collected and used.
3. ORDER_DATASET (source: Inland terminal data prepared in Gumuskaya et al. (2020a)): Each row represents a single transport order (or booking) from a shipper to/from the inland terminal from/to the port. Here, order requirements are specified such as "Origin Terminal", "Destination Terminal", "Expected Arrival Date" and "Due Date". Unlike the other two datasets, this dataset does not include actual arrival dates vis-à-vis expected arrival dates; only the day that the container is expected to arrive is known. Furthermore, the expected arrival and due dates are on day basis, while in the other datasets these fields are specified in units of seconds. The further details of the dataset can be found in Gumuskaya et al. (2020a), where it is originally collected and used.

A.2. Shortcomings of the original datasets and resolutions

The numerical experiments of this study necessitate each order to be complete with order requirements (e.g. expected arrival date, due date, origin terminal, destination terminal), container details (e.g. reefer) and actual events (e.g. actual delays on arrival). Unfortunately, none of the three datasets alone includes all of the fields. Fabricating the data from scratch could be an option, but this is not preferred because we aim to show how data driven methods can reveal patterns in real life, and how revealing those patterns may improve decision making under uncertainty. If we fabricated data, we would be creating those patterns ourselves without substantiation. For this reason, we chose joining these datasets.

Joining the datasets posed the following challenges:

1. In CONTAINER_DATASET, the delays could be calculated by taking the time difference between ETA's and ATA's but this is inconvenient for the following reason: In practice, container terminals oblige the barge operators to make call requests latest three days before the requested time of the barge call. Therefore, late ETA notifications that are made by the deep sea vessels, for example within the last day to expected arrival, is not useful for the barge operator. On top of this three-day period, inland terminal also has internal procedures such as planning container loading/unloading within the terminal and truck planning for pre/end hauling. Hence, only the ETA notifications at least four days before the expected arrival time are usable.

   This difficulty is solved by joining CONTAINER_DATASET with DEEP_SEA_VESSEL_DATASET. From DEEP_SEA_VESSEL_DATASET, the ETA's with a minimum of four days notice are imported to CONTAINER_DATASET. In CONTAINER_DATASET, the rows with imported ETA's are kept and the rest is removed. This final dataset, which we refer to as CONTAINER_POOL, consists completely of the containers with usable ETA's and ATA's. The final columns of CONTAINER_POOL are "Container ID", "Terminal", "Reefer", "Deep sea vessel call size" and " Actual delay". Note that here the "Actual delay" is the time between the ETA of the deep sea vessel to the port and the time the container is released (i.e. on the ground in the container terminal and ready to be picked up). Resulting CONTAINER_POOL dataset has 49,587 rows over a three month period.

2. CONTAINER_POOL dataset is sufficient for the first step of the solution approach, i.e. training a decision tree to predict the delays (see Section 4.2). However, for computational experiment it is not enough: For each transport order we need order requirements (i.e. origin terminal, destination terminal, expected arrival date, due date), container details to predict delays (i.e. reefer indicator, deep sea vessel call size, terminal name) and actual delay. ORDER_DATASET already includes order requirements but container details and actual delay is missing. For each order, we fill in the missing fields, namely "Reefer Indicator", "Deep sea vessel call size", "Terminal Name" and "Actual Delay", by random sampling from CONTAINER_POOL. During the random sampling for each order, we use the existing order information as much as possible. The details of how random sampling is done is explained for each column as follows, and summarized in Fig. 5:

   i. "Reefer": It shows if the container is refrigerated or not. For each order in ORDER_DATASET, we randomly select a container from CONTAINER_POOL with replacement and assign its "Reefer" value to the order.
   ii. "Deep sea vessel call size": It is important to realize that the delays of containers are not independent. For example, if a Container A and Container B are both import containers and have the same expected arrival date to the same terminal, probably they share the same deep sea vessel, which implies the vessel delay has to be the same. To represent this correlation for import containers, we identify the orders in ORDER_DATASET sharing the same expected arrival date and origin terminal with a unique "Group ID". Similarly for export containers, we identify the orders sharing the same due date and destination terminal with "Group ID". We assume such containers are from the same shippers and there is a correlation of delay for preparing the cargo. For each "Group ID" in the order set then, we randomly select a container from CONTAINER_POOL with replacement and assign the "Deep sea vessel call size" to all orders in the selected "Group ID". This same container is used in the following step (i.e. for "Terminal name") as well, to be explained shortly.
   iii. "Terminal name": ORDER_DATASET already includes the terminal names, however these terminal names do not match with those in CONTAINER_POOL. The exact matching is crucial since the DT is trained on CONTAINER_POOL. Hence, for the selected "Group ID" in the previous step, we fill in the "Terminal name" of the selected container from CONTAINER_POOL.
   iv. "Actual Delay": Until now, the container details are completed that are necessary for making predictions. In the simulation stage, "Actual delay" is necessary to evaluate the actual handlings and costs, which is subject to randomness in line with container details. For this reason, while filling in the "Actual Delay" we make another sampling: we check "CONTAINER_POOL" and determine which containers have the same "Reefer Indicator", "Deep sea vessel call size" and "Terminal name". Out of this subset, we select a random "Actual Delay" with replacement and assign to the order. The reason of sampling from this subset rather than sampling from the whole CONTAINER_POOL is as follows: the whole point of decision tree is to reveal patterns between the features (i.e. "Reefer Indicator", "Deep sea vessel call size" and "Terminal") and the predicted class (i.e."Actual Delay") in CONTAINER_POOL. If we did not do the subsetting, we would destroy any existing pattern between the predicted (i.e. actual delay) and features (i.e. Terminal, reefer, deep sea vessel call size), which would make the predictions unfairly inaccurate.

In conclusion, we reach a dataset with 42,355 orders with columns as in Fig. 5, where all of the required fields (i.e. order requirements, container details and actual delays) are present. Among these fields, "ACTUAL DELAY" needs further elaboration. Originally, the ETA's and ATA's of deep sea vessels are recorded up to the details of seconds in CONTAINER_DATASET and DEEP_SEA_VESSEL_DATASET. Such precision is not meaningful under the high uncertainty in deep sea vessel arrivals. Often, the container terminals change call times with short notice to earlier or later times. Furthermore, the barge operator makes plans on daily basis. This compels a need for converting the delays to days. This conversion is executed as follows: we allow margin of delays and set a 6 h threshold for the delays. If the container is ready to be picked up within 6 h of the ETA, then the delay is 0. So the container can be picked up in the expected arrival date or later. If the delay is greater than 6 h but less than 30 h, then the delay is 1 day and thus the container can be picked up earliest one day after expected arrival date. This conversion and the distribution of delays is summarized in Table 6.

By using the container details, we will make predictions on the probability distribution of actual delays, and use these predictions to improve the decision making in the 2-stage stochastic MIP.

A.3. The preliminary runs to determine the number of scenarios

The computational times and the value of objective function values of the solutions are depicted in Figs. 6 and 7 for two arbitrarily selected periods, Period 0 and 40.

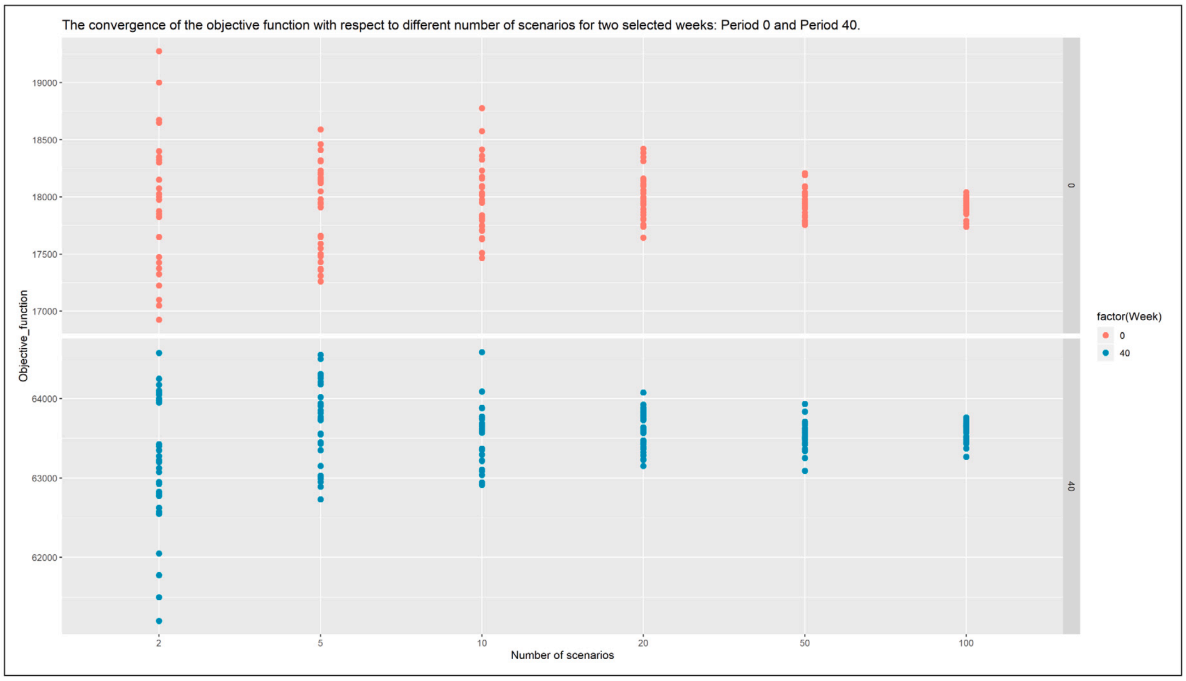


Fig. 6. The objective function values of the stochastic programs against the number of scenarios.

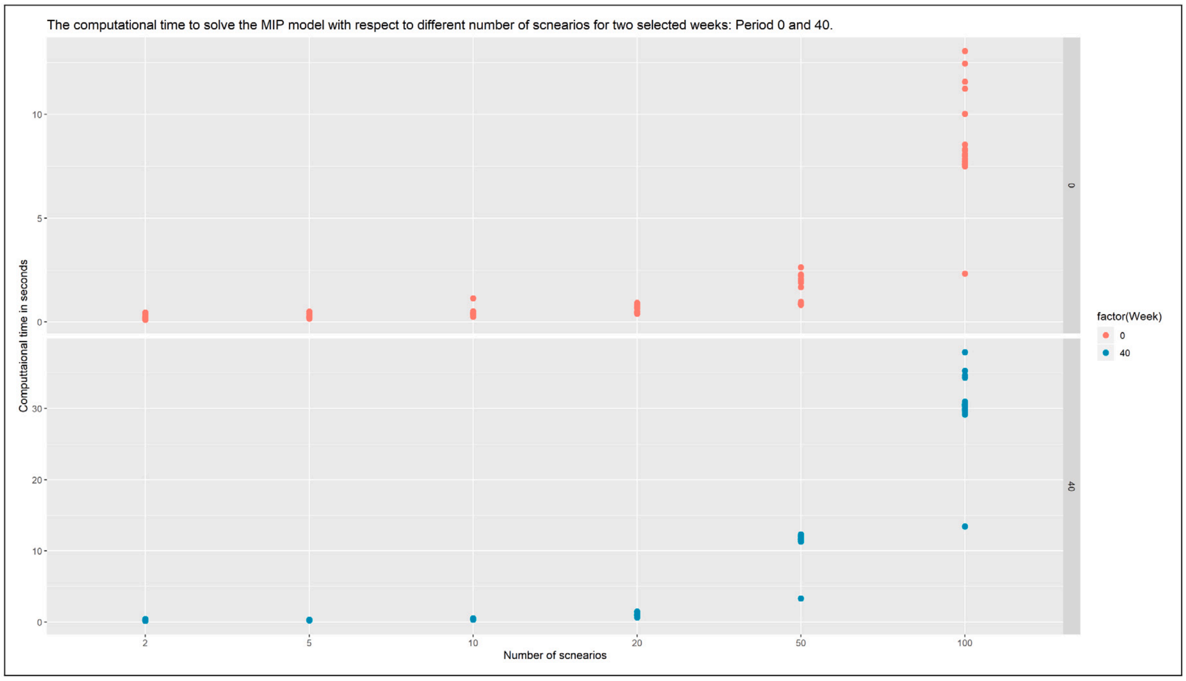


Fig. 7. The computational times of solving the stochastic program for one period against the number of scenarios.

References

Balster, A., Hansen, O., Friedrich, H., Ludwig, A., et al., 2020. An ETA prediction model for intermodal transport networks based on machine learning. Bus. Inf. Syst. Eng.: The Int. J. WIRTSCHAFTSINFORMATIK 1–14.

Barbour, W., Martinez Mori, J.C., Kuppa, S., Work, D.B., 2018. Prediction of arrival times of freight traffic on US railroads using support vector regression. Transp. Res. C 93, 211–227. http://dx.doi.org/10.1016/j.trc.2018.05.019, URL http://www.sciencedirect.com/science/article/pii/S0968090X18307101.

Bello, I., Pham, H., Le, Q.V., Norouzi, M., Bengio, S., 2016. Neural combinatorial optimization with reinforcement learning. arXiv preprint arXiv:1611.09940.

Bertsimas, D., Chang, A., Rudin, C., 2011. Ordered rules for classification: A discrete optimization approach to associative classification. In: SUBMITTED TO the ANNALS of STATISTICS. Citeseer.

Bertsimas, D., Gupta, V., Kallus, N., 2018. Data-driven robust optimization. Math. Program. 167 (2), 235–292.

Bhattacharya, A., Kumar, S.A., Tiwari, M., Talluri, S., 2014. An intermodal freight transport system for optimal supply chain logistics. Transp. Res. C 38, 73–84. http://dx.doi.org/10.1016/j.trc.2013.10.012, URL http://www.sciencedirect.com/science/article/pii/S0968090X13002271.

Casey, M.S., Sen, S., 2005. The scenario generation algorithm for multistage stochastic linear programming. Math. Oper. Res. 30 (3), 615–631.

Chen, Y., Yuan, Z., Chen, B., 2018. Process optimization with consideration of uncertainties—An overview. Chin. J. Chem. Eng. 26 (8), 1700–1706.

Crespo-Vazquez, J.L., Carrillo, C., Diaz-Dorado, E., Martinez-Lorenzo, J.A., Noor-E-Alam, M., 2018. A machine learning based stochastic optimization framework for a wind and storage power plant participating in energy pool market. Appl. Energy 232, 341–357.

Dogan, N., Tanrikulu, Z., 2013. A comparative analysis of classification algorithms in data mining for accuracy, speed and robustness. Inf. Technol. Manag. 14 (2), 105–124.

Douma, A., Schutten, M., Schuur, P., 2009. Waiting profiles: An efficient protocol for enabling distributed planning of container barge rotations along terminals in the port of Rotterdam. Transp. Res. C 17 (2), 133–148.

Fazi, S., Fransoo, J.C., Van Woensel, T., 2015. A decision support system tool for the transportation by barge of import containers: a case study. Decis. Support Syst. 79, 33–45.

Fazi, S., Roodbergen, K.J., 2018. Effects of demurrage and detention regimes on dry-port-based inland container transport. Transp. Res. C 89, 1–18. http://dx.doi.org/10.1016/j.trc.2018.01.012.

Guevara, E., Babonneau, F., Homem-de Mello, T., Moret, S., 2020. A machine learning and distributionally robust optimization framework for strategic energy planning under uncertainty. Appl. Energy 271, 115005.

Gumuskaya, V., van Jaarsveld, W., Dijkman, R., Grefen, P., Veenstra, A., 2020b. A framework for modelling and analysing coordination challenges in hinterland transport systems. Accepted: Maritime Economics and Logistics 22, Special Issue: Port-Hinterland Transport and Logistics - Emerging Trends and Frontier Research. http://dx.doi.org/10.1057/s41278-019-00139-1.

Gumuskaya, V., van Jaarsveld, W., Dijkman, R., Grefen, P., Veenstra, A., 2020a. Dynamic barge planning with stochastic container arrivals. Transp. Res. Part E: Logist. Transp. Rev. 144, 102161. http://dx.doi.org/10.1016/j.tre.2020.102161.

Hottung, A., Tanaka, S., Tierney, K., 2020. Deep learning assisted heuristic tree search for the container pre-marshalling problem. Comput. Oper. Res. 113, 104781. http://dx.doi.org/10.1016/j.cor.2019.104781, URL http://www.sciencedirect.com/science/article/pii/S0305054819302230.

Hu, M.-C., Hobbs, B.F., 2010. Analysis of multi-pollutant policies for the US power sector under technology and policy uncertainty using MARKAL. Energy 35 (12), 5430–5442.

Huang, Y.-H., Wu, J.-H., Hsu, Y.-J., 2016. Two-stage stochastic programming model for the regional-scale electricity planning under demand uncertainty. Energy 116, 1145–1157.

Irannezhad, E., Prato, C.G., Hickman, M., 2020. An intelligent decision support system prototype for hinterland port logistics. Decis. Support Syst. 130, 113227.

Kaut, M., Wallace, S.W., 2007. Evaluation of scenario-generation methods for stochastic programming. Pac. J. Optim. 3 (2), 257–271.

Lium, A.-G., Crainic, T.G., Wallace, S.W., 2009. A study of demand stochasticity in service network design. Transp. Sci. 43 (2), 144–157.

Lodi, A., Zarpellon, G., 2017. On learning and branching: a survey. Top 25 (2), 207–236.

Löhndorf, N., 2016. An empirical analysis of scenario generation methods for stochastic optimization. European J. Oper. Res. 255 (1), 121–132.

Mak, W.-K., Morton, D.P., Wood, R., 1999. Monte Carlo bounding techniques for determining solution quality in stochastic programs. Oper. Res. Lett. 24 (1), 47–56. http://dx.doi.org/10.1016/S0167-6377(98)00054-6, URL http://www.sciencedirect.com/science/article/pii/S0167637798000546.

Mitra, S., Domenica, N.D., 2010. A review of scenario generation methods. Int. J. Comput. Sci. Math. 3 (3), 226–244.

Mitra, S., Lim, S., Karathanasopoulos, A., 2019. Regression based scenario generation: Applications for performance management. Opera. Res. Perspect. 6, 100095. http://dx.doi.org/10.1016/j.orp.2018.100095, URL https://www.sciencedirect.com/science/article/pii/S2214716018300940.

Pflug, G.C., Pichler, A., 2016. From empirical observations to tree models for stochastic optimization: convergence properties. SIAM J. Optim. 26 (3), 1715–1740.

Powell, W.B., 2019. A unified framework for stochastic optimization. European J. Oper. Res. 275 (3), 795–821.

Ridier, A., Chaib, K., Roussy, C., 2016. A dynamic stochastic programming model of crop rotation choice to test the adoption of long rotation under price and production risks. European J. Oper. Res. 252 (1), 270–279.

van Riessen, B., Negenborn, R.R., Dekker, R., 2016. Real-time container transport planning with decision trees based on offline obtained optimal solutions. Decis. Support Syst. 89, 1–16.

Rivera, A.E.P., Mes, M.R., 2017. Anticipatory freight selection in intermodal long-haul round-trips. Transp. Res. Part E: Logist. Transp. Rev. 105, 176–194.

Şafak, Ö., Çavuş, Ö., Aktürk, M.S., 2018. Multi-stage airline scheduling problem with stochastic passenger demand and non-cruise times. Transp. Res. B 114, 39–67.

Shapiro, A., Dentcheva, D., Ruszczyński, A., 2014. Lectures on stochastic programming: modeling and theory. SIAM.

Smirnov, D., Huchzermeier, A., 2020. Analytics for labor planning in systems with load-dependent service times. European J. Oper. Res. 287 (2), 668–681.

Smulders, F., 2019. The value of digital platforms for transportation planning in the port of rotterdam. Master Thesis. Repository of Eindhoven University of Technology.

SteadieSeifi, M., Dellaert, N., Nuijten, W., Woensel, T.V., Raoufi, R., 2014. Multimodal freight transportation planning: A literature review. European J. Oper. Res. 233 (1), 1–15. http://dx.doi.org/10.1016/j.ejor.2013.06.055, URL http://www.sciencedirect.com/science/article/pii/S0377221713005638.

Tulabandhula, T., Rudin, C., 2014. Robust optimization using machine learning for uncertainty sets. arXiv preprint arXiv:1407.1097.

Vinyals, O., Fortunato, M., Jaitly, N., 2015. Pointer networks. In: Advances in Neural Information Processing Systems. pp. 2692–2700.

Zhang, C., Liu, C., Zhang, X., Almpanidis, G., 2017. An up-to-date comparison of state-of-the-art classification algorithms. Expert Syst. Appl. 82, 128–150.

Zweers, B.G., Bhulai, S., Mei, R.D., 2020. Planning hinterland container transportation in congested deep-sea terminals. Flex. Serv. Manuf. J. 1–40.