Maintenance location routing for rolling stock

Research School for Operations Management and Logistics

Maintenance Location Routing for Rolling Stock: Robust and Stochastic Programming Formulations

D.D. Tönissen, J.J. Arts, Z.J.M. Shen

Beta Working Paper series 521

BETA publicatie	WP 521 (working paper)
ISBN	
ISSN	
NUR	
Eindhoven	January 2017

Maintenance Location Routing for Rolling Stock: Robust and Stochastic Programming Formulations

Denise Tönissen, Joachim Arts

School of Industrial Engineering, Eindhoven University of Technology, Eindhoven, The Netherlands

Zuo-Jun Max Shen

Department of Industrial Engineering and Operations Research, University of California,

Berkeley, California 94720

Abstract

Rolling stock needs regular maintenance in a maintenance facility. Rolling stock from different fleets needs to be routed to maintenance facilities using interchanges between train lines and possible empty drives. We consider the problem of locating maintenance facilities in a railway network under uncertain or changing line planning, fleet planning and other factors. These uncertainties and changes are modeled by a discrete set of scenarios. We show that this new problem is NP-hard and provide a two-stage stochastic programming and a two-stage robust programming formulation. The second stage decision is a maintenance routing problem with similarity to a minimum cost-flow problem. We prove that the facility location decisions remain unchanged under a simplified routing problem and this gives rise to an efficient mixed integer programming (MIP) formulation. We also provide an accelerated Benders decomposition algorithm that uses these insights and bounds obtained from this MIP formulation. This result also allows us to find an efficient decomposition algorithm for the robust programming formulations based on Scenario Addition (SA). Computational work on instances of industrial size and larger shows that our improved MIP formulation outperforms Benders decomposition in computational time. SA improves the computational time for the robust formulation even further and can handle larger instances due to more efficient memory usage. Finally, we apply our algorithms to a case study at the Dutch Railways.

Keywords: Two-stage robust optimization, two-stage stochastic optimization, row-and-column generation, facility location, rolling stock, maintenance routing

1 Introduction

In this paper, we study the optimal location of maintenance facilities for rolling stock in a railway network. Like most facility location problems, we have a set of candidate facilities and their

costs, and we have to decide which facilities to open. However, the *maintenance location routing problem* has essential features that makes it substantially different from other facility location models. In particular, the customers of facilities (rolling stock) have to travel to a facility over a fixed railway network. There are multiple *train lines*, a path in the railway network operated with a certain frequency, and the preferred way for train units to enter a maintenance facility is to interchange from one train line to another until a maintenance facility is reached. Whether such an interchange is possible depends on the line-planning in the railway network, the shunting infrastructure at locations where two or more lines intersect, and the rolling stock schedule. Sometimes it is necessary to plan an empty train drive to reach a maintenance facility. We call the problem of routing a train unit to a maintenance facility the maintenance routing problem (MRP). This routing problem cannot be separated from the facility location problem because whether a facility is easy to reach depends intricately on the railway infrastructure and line planning.

The next essential feature of maintenance facility location problems is that the line planning within a railway network changes regularly to accommodate changing travel demands. Any reasonable facility location plan must work well under a wide variety of line planning scenarios. This includes changes in how lines run, up and down-scaling of service frequencies on any given line, the rolling stock types assigned to the lines, and the introduction of new rolling stock types.

To deal with the features we outline above (maintenance location routing and line planning), we provide two different and novel models to help managers decide where to locate maintenance facilities. The first model is a two-stage robust programming model, where we minimize the cost for the worst line planning scenario. We call this the robust maintenance location routing problem (RMLRP). The second model seeks to minimize the annual cost of operating facilities and routing trains to facilities averaged over a set of line planning scenarios. This model is therefore called the stochastic maintenance location routing problem (SMLRP).

The first stage decision for the R/SMLRP is to open or close a facility, given a set of candidate facilities, deterministic facility costs, and a discrete set of line planning scenarios. The second stage decision is taken once a scenario is realized, and corresponds with the MRP. The MRP is modelled as a linear programming problem, which has some similarity to the minimum-cost flow problem. We show that the two-stage models can be reformulated to mixed integer programming models, and prove that exactly the same facilities are opened when a simplified routing problem is used. Simplifying the routing problem within the combined problem, decreases the number of variables by the number of facilities times the number of train lines raised to the square for every scenario. However, after solving the improved mixed integer program (IMIP), the MRP should be solved for every scenario for the opened facilities, to recover the routings and the assignments. As the MRP is a linear programming problem, this can can be done in polynomial time. We show that using this IMIP, instances can be solved quickly by CPLEX and are solved a lot faster than the two-stage models by accelerated Benders decomposition (Benders, 1962; van Slyke and Wets, 1969). Furthermore, we improve the computational time for the RMLRP

further, by providing a scenario addition (SA) method, which is a row-and-column (also called column-and-constraint) generation method (similar to Zeng and Zhao, 2013) for the RMLRP. SA adds the scenario constraints iteratively to the IMIP until an optimal solution is found, which improves the computational time and memory requirements for most instances.

The main contributions of this paper are:

1. We are the first to study a maintenance location problem on a railway network. In this setting it is necessary to incorporate the maintenance routing of train units in the facility location decision. This leads to a new class of problems that we call maintenance location routing problems (MLRP).

2. We provide a two-stage stochastic (SMLRP-2SSO) and robust programming formulation (RMLRP-2SRO) for the MLRP that can deal with uncertainties that are inherent in any application. In particular, our model can deal with uncertainties in line planning and rolling stock management.

3. We provide an efficient mixed integer programming formulation (which we call IMIP), that decreases the number of variables by the number of facilities times the number of train lines raised to the square for every scenario relative to a standard MIP formulation. The IMIP can solve both the SMLRP and RMLRP for industrial size instances. This formulation outperforms accelerated Benders decomposition computationally.

4. We provide a SA algorithm (a row-and-column generation method) that can solve the RMLRP even more efficiently than the IMIP for instances with many scenarios.

5. We showcase how our algorithms can be used in a case study at the Dutch railways.

The paper starts with a literature review, followed by a more detailed description of the maintenance routing problem. In Section 4, we model the RMLRP and the SMLRP as two-stage problems, and in Section 5, we reformulate the problems as a mixed integer programming problem. In Section 6, we present an accelerated benders decomposition algorithm and our scenario addition method. In Section 7, we do experiments on randomly generated instances. We research the influence of the number of scenarios and facilities on the solution time and compare the algorithms. In Section 8, we do a case study with data from the Dutch railways, and show that our algorithms can be used in practice.

2 Literature Review

Facility location models have been studied extensively in the literature (Daskin, 1995; Drezner and Hamacher, 2001). The combination with operational supply chain decisions (Melo et al., 2009), vehicle routing (Nagy and Salhi, 2007), and uncertainty have also been studied in the literature (Snyder, 2006). However, the combination of facility location with maintenance routing

has only been studied in the context of aviation applications (Feo and Bard, 1989; Gopalan, 2014). Furthermore, to the best of our knowledge, there are only three papers that combine maintenance with facility location (Lieckens et al., 2013; Rappold and Van Roo, 2009; van Ommeren and Bumb, 2006). However, the settings of these papers differ considerably from ours, and the authors use heuristics while we seek optimal solutions. We first describe the maintenance routing literature and we continue to the two-stage robust and stochastic facility location literature.

2.1 Maintenance Routing

Many papers have been written about maintenance routing for railway applications. Most papers consider it to be part of the medium or long-term vehicle routing problem. Anderegg et al. (2003), consider the situation in the German and Swiss Federal Railway and use a minimum cost flow formulation that is often used for vehicle routing. Maintenance cannot be adapted into the flow model and a heuristic modification is used to satisfy the maintenance constraints. If possible, they use a local fix that makes it possible to do maintenance directly in the train unit rotation. For the train units where this cannot be done, the train unit is interchanged with a train unit that is already at the maintenance location and continues the routing of the train unit that requires maintenance such that the routing cycle remains intact. The authors' goal is to use as few spare train units as possible.

Maróti and Kroon (2005, 2007) consider maintenance routing for the Dutch railways, where maintenance routing is not part of the medium or long-term vehicle scheduling problem. They use a two to five days time window during which the train unit is routed to the maintenance facility. The main reason for this, is that timetables and rolling stock schedules are dense, and that there are many disturbances. Consequently, long term models cannot take shunting issues and disturbances into account. The author model this problem with multicommodity flow models, and they solve the NP-hard models with CPLEX. Other papers consider variants of the locomotive planning problem (LPP). The locomotive planning problem assigns locomotives to a set of train units in such a way that it minimizes the cost and satisfies a number of business and operational constraints. For a recent survey on the locomotive assignment problem, that also includes variants with some maintenance constraints, see Piu and Speranza (2014). Furthermore, there are many papers about maintenance routing for aviation such as Clarke et al. (1997), Talluri (1998), and Sarac et al. (2006).

2.2 Two-stage Robust and Stochastic Facility Location

Uncertainty for facility location models can be classified in three categories (Shen et al., 2011): receiver-side uncertainty, in-between uncertainty, and provider-side uncertainty. Like most stochastic facility location models (see the references in Snyder (2006) and Swamy and Shmoys (2006)), our paper focuses on the first two uncertainties. These uncertainties are related to customer uncertainty (for example customer demand, customer location) and incomplete knowledge about

the transportation network topology, transportation times or costs between facilities and customers.

A common feature of the receiver-side and in-between uncertainties is that the uncertainty does not change the topology of the provider-receiver network once the facilities have been built. Our problem does not share this feature as our scenarios do change the topology of the provider-receiver network; a different line planning changes the network over which rolling stock is routed to a maintenance facility. Consequently, the routing and assignment of maintenance visits to the facilities is different for every scenario.

The SMLRP has a lot of similarity with supply chain network design under uncertainty, which includes the location of facilities within a supply chain. In Santoso et al. (2005) a supply chain network consisting of suppliers, processing facilities and customers has to be designed under cost, demand, supply and capacity uncertainty. A sample average approximation is used to generate a discrete set of scenarios and the model is solved with Benders decomposition. Santoso et al. (2005) describe and test many acceleration techniques, that we also implement in this paper. The accelerated Benders decomposition method works well for their problem, but is less successful for our model as compared to the IMIP. Other examples of Benders decomposition for supply chain network design include Costa (2005) (fixed charge network design survey), Üster and Agrahari (2011) (freight-forwarding network design), Santibanez-Gonzalez and Diabat (2013) (reverse supply chain design), and Khatami et al. (2015) (closed loop supply chain network design).

Two type of methods are generally used for two-stage robust problems. The first method is similar to Benders decomposition and uses constraint generation based on the dual information of the slave problem. Álvarez-Miranda et al. (2015), use Benders decomposition accelerated with additional cuts and a primal heuristic on a two-stage robust facility location problem with a discrete set of scenarios. An important difference with our paper is that their allocation of customers to facilities does not include the routing of customers (rolling stock) to (maintenance) facilities. Another difference is the considered uncertainty and recoverability of the model. In our model, we open facilities in the first stage and allocate customers to these facilities in the second stage. Álvarez-Miranda et al. (2015) open facilities and assign customers in the first stage and the second stage is used to recover the facilities and customer assignment to the revealed scenario. Gabrel et al. (2014) solve a non-linear convex two-stage robust location transportation problem. In this problem a commodity has to be transported from each of the m potential sources to each of the n destinations. The demand of the destinies is uncertain. The authors use a cutting plane algorithm based on Benders decomposition, where the slave problem is already NP-hard. Furthermore, Benders decomposition has been applied to two-stage robust unit commitment problems (Bertsimas et al., 2013; Jiang et al., 2011).

The second method consist of row-and-column generation procedures, also known as column and constraint generation. Zeng and Zhao (2013) shows that a row-and-column generation procedure performs an order of magnitude faster than benders decomposition for a two-stage

robust location transportation problem with demand levels in a polyhedral uncertainty set. Chan et al. (2015) apply a row-and-column generation procedure to a robust facility location problem where the demand points are uncertain and An et al. (2014) apply it to the reliable p-median facility location problem. The strategy is also used for two-stage robust unit commitment (An and Zeng, 2015; Zhao and Zeng, 2012) and a two-stage robust distribution network reconfiguration problem (Lee et al., 2015).

3 Maintenance Routing

The goal of the maintenance routing problem is to find the optimal routing for train units to enter given maintenance facilities with given capacities. A *maintenance facility* is a facility that is responsible for the planned inspections and maintenance of rolling stock. The frequency of inspections and maintenance is dependent on the rolling stock type and typically occurs once every half year up to every month. The maintenance routing is influenced by the current *line plan*, a set of routes (paths) in a network of rails, operated with a certain frequency by a specific rolling stock type. The stations where a line starts or ends are called *end stations*, and all passengers leave the train and the train unit drives back or continues on another line after a break. Between the end stations, a line often has many other regular train stations where passengers can leave or enter the train. The transport from the train lines to the maintenance facilities is done with help of *interchanges*. An interchange is swapping the destinations of two train units of the same rolling stock type which are at connecting train lines. Connecting train lines share an end station and at that end station the train unit can be interchanged. The train units continue at each others train line after such an interchange. A train that requires maintenance is interchanged with another train unit until it reaches a train line connected to a maintenance facility. When a maintenance facility cannot be reached by a specific train unit via these interchanges, *dead heading* is used for the remaining trip. Dead heading is driving with an empty train (no passengers), and is undesirable. Dead heading gives additional driving cost and the train unit is not available for public transport, which can result in shorter trains and passengers discomfort.

The operational problem of routing specific train units to their maintenance facilities is studied in Maróti and Kroon (2005, 2007). In this section, we introduce a new maintenance routing model that operates on an aggregate level. This enables us to combine this model with facility location decisions later. Our objective is to minimize the average annual interchange and dead-heading cost. Consequently, we do not consider separate train units, but work with the average number of maintenance visits departing from each line per year.

3.1 Problem Description

Given is a physical rail network $G_P = (N_P, E_P)$, consisting of rails E_P, all stations N_P, all *end stations* $S \subseteq N_P$ and a set with opened facilities $O \subseteq S$, with for every opened facility a capacity, that represents the maximum number of maintenance visits the location can handle per year.

Furthermore, we are given a line plan, which consist of a set of *lines* L, with for each line the type of rolling stock, the maintenance frequency per year, its end stations and the dead heading cost to each maintenance facility. The line plan also specifies the set of possible interchanges, with a coordination cost for each interchange, end station interchange capacities and an interchange budget.

The annual number of interchanges in the network is restricted by a budget G. The interchange capacity highly depends on the frequently changing rolling stock schedule and it is possible to improve certain interchanges by making small changes to the rolling stock schedule. Modelling this interchange budget allows us to gain insight into which interchanges should be made possible or improved, while also constraining the number of interchanges that can be used. Furthermore, we constrain the annual number of interchanges at each end station. The objective is to minimize the average interchange and deadheading cost per year.

In Figure 1, we show a physical network graph on the left. In the physical network graph, we exclude all stations that are not an end station in at least one of the shown line planning possibilities for ease of exposition. The number of stations in a network of this size can easily be 50 or more. In the middle and the right of Figure 1, we show line planning possibilities for this network. For the MRP, only one line plan for the network G_P is given. However in Section 4 the different line plan possibilities for network G_P play an important role. Each line (edge) connects one end station (node) to the other. There are two train types in this example. Train type 0, is a regional train, stopping at every small station, while type 1 is an intercity train that only stops at the large cities. An example of an interchange for the middle picture is line (U, V) to line (V, W), while an interchange from (U, V) to line (U, W) is not possible because the train types do not match.

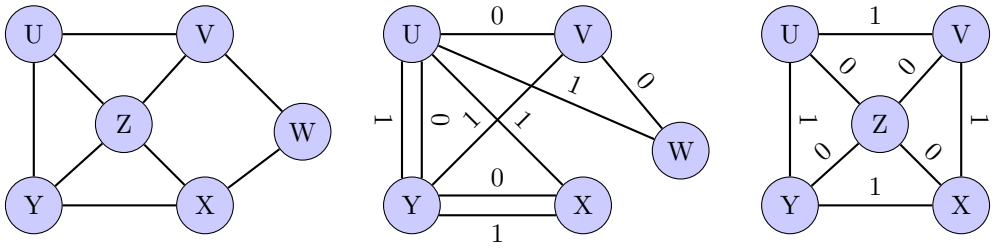


Figure 1: The physical rail network on the left and two line planning possibilities

3.2 Maintenance Routing Model

We model the MRP as a flow model. We only allow interchanges followed by dead heading directly to the maintenance facility. The reason for this is practical, and not a restriction of the model. Routes with deadheading followed by interchanges are not often used in practice, can be very expensive, and cause imbalances in the number of train units per line, which need

to be solved. Consequently, we do not include these kind of routes in our model. We apply the following transformations to the line planning graph, to create the maintenance routing flow graph $G_F = (N_F, A_F)$:

- For every line, we make a node, the set with these nodes is denoted by N_L.
- We create one source $\mathcal{S}$ that is connected with a directed arc to each node in N_L.
- We create an arc between lines where an interchange is possible, with as cost the interchange coordination cost. The set of these interchange arcs is denoted A_I.
- We create a node for every open facility and denote the set of open facilities by N_O. Each node in N_O is connected with an arc to the sink $\mathcal{T}$.
- For each node $n \in N_L$, we make an arc to each facility. The cost of this arc is the dead heading cost of the line to the facility. The cost of the arc is 0, when dead heading is not necessary because the line associated with the node is connected to the facility. The set of these incoming facility arcs is denoted by A_O.

In Figure 2, we demonstrate how to apply this transformation to make a graph for the first line planning graph (the middle one in Figure 1). We assume that facilities $N_O = \{U, W, Y\}$ are opened. The left figure depicts the line planning graph, with the name of the lines and the end stations, the right figure depicts the flow network that is used as input for the MRP.

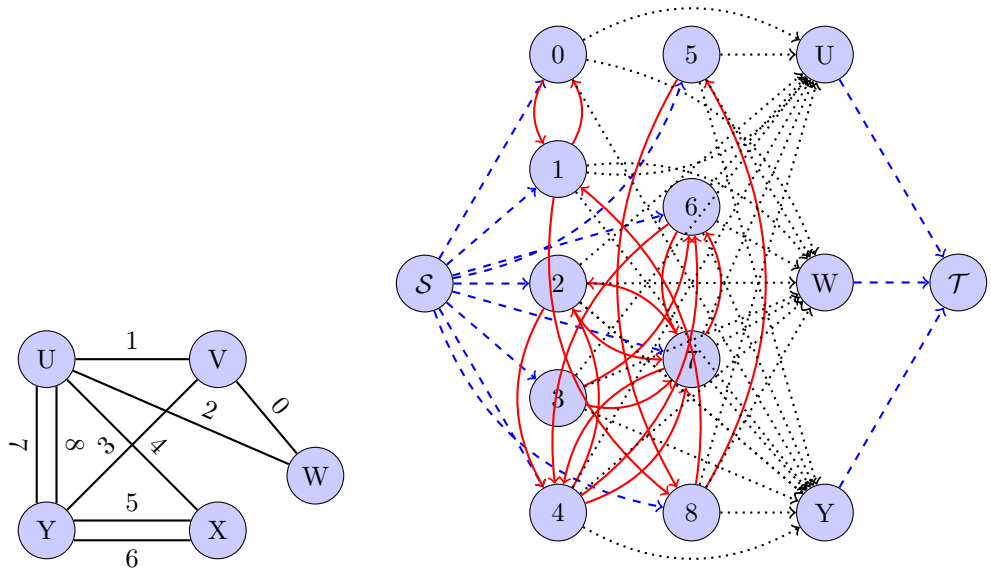


Figure 2: Left a line planning possibility and right the resulting flow graph. The arcs from and to the source and sink are dashed blue, the interchange arcs solid red and the arcs to the facilities are dotted.

The aforementioned additional combinations of interchanges and deadheading can easily be allowed by creating an additional arc from every line node to every line node, which represents the dead heading from one line to another.

Proposition 1. *The number of nodes and arcs in the flow graph is polynomial in the number of lines and end stations.*

Proof. The number of nodes in the flow graph is equal to $|L| + |N_O| + 2$. The number of arcs is equal to $|L| + |L||N_O| + |A_I| + |N_O|$, where $|A_I|$ is bounded by $|L|^2$. □

The flow through arc $a \in A_F$ associated with the yearly maintenance frequency originating from line $l \in L$, is represented by the decision variable $z_l(a)$. For example $z_4(2,7)$ represents the frequency of interchanges from line 2 to line 7 for a maintenance visit originating from line 4, and $z_1(8,y)$ represents the frequency of maintenance visits originating from line 1, that reach maintenance facility y via line 8. We define $\delta_{\text{in}}(n)$ and $\delta_{\text{out}}(n)$ as the set of ingoing and outgoing arcs of node $n \in N_F$. The cost of arc a of type l is $c_l(a)$, which is only defined for arcs in the set $A_I \cup A_O$. Node n_l is the node associated with line l, and the maintenance frequency per year for this node is m_l. The end station interchange capacity g_s, is a capacity on the number interchanges at end station $s \in S$. The set of arcs representing the interchanges going through end station s is defined as A_s. Furthermore, G is the interchange budget. For the facilities, q_n $\forall n \in N_O$ denotes the capacity of facility n in terms of the possible yearly frequency of maintenance visits to this facility. We formulate the following linear programming model:

$$\text{MRP} = \min \sum_{a \in A_I \cup A_O} \sum_{l \in L} c_l(a) z_l(a)$$

Subject to:

$$\sum_{a \in \delta_{\text{in}}(n)} \sum_{l \in L} z_l(a) \leq q_n \qquad \forall n \in N_O, \tag{1}$$

$$\sum_{a \in \delta_{\text{in}}(n)} z_l(a) = \sum_{a \in \delta_{\text{out}}(n)} z_l(a) \qquad \forall n \in N_F \setminus \{\mathcal{S}, \mathcal{T}\}, \quad \forall l \in L, \tag{2}$$

$$z_l(a) = m_l \qquad \forall l \in L, \quad a \in \delta_{\text{in}}(n_l), \tag{3}$$

$$\sum_{a \in \delta_{\text{in}}(\mathcal{T})} \sum_{l \in L} z_l(a) = \sum_{l \in L} m_l, \tag{4}$$

$$\sum_{a \in A_s} \sum_{l \in L} z_l(a) \leq g_s \qquad \forall s \in S, \tag{5}$$

$$\sum_{a \in A_I} \sum_{l \in L} z_l(a) \leq G, \tag{6}$$

$$z_l(a) \geq 0 \qquad \forall a \in A_F, \quad \forall l \in L. \tag{7}$$

We can solve the MRP in polynomial time because it is a linear program. Constraints (1) restrict the number of yearly maintenance visits that can be assigned to opened facility n to its

capacity. Constraints (2) are the flow conservation constraints, while Constraints (3) and (4) guarantee that every maintenance visit is assigned to a facility. Constraints (4) is necessary to exclude alternative solutions with more flow than the number of yearly maintenance visits; these solutions are possible because some of the routes to the facilities have zero costs. Constraints (5) and (6) are the end station and budget interchange capacity constraints.

The problem is similar to the minimum cost flow problem with the exception of the multiple commodities $l \in L$ and Constraints (1), (5) and (6). We show that even with these additional constraints the MRP has an integer solution $z_l(a)$ if g_s $(\forall s \in S)$, G, m_l $(\forall l \in L)$ and q_n $(\forall n \in N_F)$ are integer.

Proposition 2. *If g_s $(\forall s \in S)$, G, m_l $(\forall l \in L)$ and q_n $(\forall n \in N_F)$ are integer, then for any basic feasible solution of the MRP each flow $z_l(a)$ $(\forall a \in A_F, \forall l \in L)$ is integer.*

The proof of Proposition 2 can be found in Appendix A.

4 Maintenance Location Routing

In this section we introduce the stochastic and robust maintenance location routing problem (SMRLP and RMRLP). We model both problems as two-stage models. We use a discrete scenario set D, in which every scenario defines a line planning (see also Figure 1 from Section 3). The first-stage decision is the decision to open or close a facility, given a set of candidate facilities, deterministic facility costs and a discrete set of scenarios. The second-stage decision is taken once a scenario is realized, and corresponds with the maintenance routing problem discussed in Section 3. The RMLRP optimizes the cost for the worst scenario $d \in D$, and the SMLRP optimizes the cost for the average case.

The R/SMLRP are NP-hard because the capacitated facility location problem (CFLP) is a special case of these problems. In the CFLP, a set of demand points, a set of facilities, the capacities of the locations and their cost, and the cost of assigning a demand point to a facility are given. The objective is to open a set of facilities and to assign each demand point to a facility, while not exceeding the capacity and minimizing the cost. Our model extends the CFLP, by including multiple discrete scenarios and a maintenance routing problem. When our model has only one scenario and the interchange budget for this scenario is 0, then only dead heading from the lines to the locations is possible. When we interpret the lines as demand points, with as demand the required maintenance visit frequency, then the deadheading cost is exactly the same as the assignment cost, and the S/RMLRP can be used to solve the CFLP.

4.1 The Models

As in Section 3, we are given a physical rail network $G = (N_P, E_P)$, consisting of rails E_P and all possible stations N_P. Furthermore, we are given a set of discrete scenarios $d \in D$, in which each scenario defines a line system: a set of lines L^d $\forall d \in D$, with for each line the type of

rolling stock and the maintenance frequency per year, the end stations $S^d \subseteq N_P \ \forall d \in D$, and their location in the physical rail network. A line plan for a scenario specifies the set of possible interchanges, with a coordination cost for each interchange, the dead heading cost from each line to each facility, the end station interchange capacities and the interchange budget. Finally we are given a set of candidate facilities $C \subseteq \bigcup_{d \in D} S^d \subseteq N_P$, with for every candidate facility a capacity, that represents the maximum yearly maintenance visit frequency the location can handle. The first stage decision is represented by the binary decision variable $y \in \{0,1\}^C$, which is 1 when a facility is opened and 0 otherwise.

For every first stage decision $y \in \{0,1\}^C$ and scenario $d \in D$, there is a maintenance flow routing graph $G_F^{dy} = (N_F^{dy}, A_F^{dy})$. The notation of the variables and parameters is similar to those in Section 3. The flow through arc a associated with the yearly maintenance frequency from line $l \in L^d$, in scenario $d \in D$, is represented by the decision variable $z_l^d(a)$. We define $\delta_{\text{in}}^d(n)$ and $\delta_{\text{out}}^d(n)$ as the set of ingoing and outgoing arcs of node n for scenario d. The cost of arc a for line l and scenario d is $c_l^d(a)$. The arc costs are only defined for the arcs in the set $\bigcup_{d \in D}(A_I^d \cup A_C^d)$. A_I^d is the set of interchange arcs, and $A_C^d = \bigcup_{n \in N_C} \delta_{\text{in}}^d(n)$, the set of arcs going to the candidate facilities. Node n_l^d is the node associated with line l for scenario d, and the maintenance frequency per year for this node is m_l^d. The end station interchange capacity g_s^d, is a capacity on the interchange frequency at end station $s \in S^d$ for scenario $d \in D$. The set of arcs representing the interchanges going through end station s for scenario d is defined as A_s^d. Furthermore, we have G^d, which is the interchange budget in scenario d. For the facilities, we define $q_n \ \forall n \in N_C$ as the capacity of facility n in terms of the possible yearly frequency of maintenance visits at this facility. The probability that scenario $d \in D$ occurs, p_d, is only defined and used for the SMLRP. We formulate the following two-stage models:

$$\text{(RMLRP-2SRO)} \quad \min \sum_{n \in N_C} c_n y_n + \max_{d \in D} \text{MRP}_d(y)$$

$$\text{(SMLRP-2SSO)} \quad \min \sum_{n \in N_C} c_n y_n + \sum_{d \in D} p_d \text{MRP}_d(y)$$

Subject to:

$$\sum_{n \in N_F} y_n q_n \geq \max_{d \in D} \sum_{l \in L^d} m_l^d, \tag{8}$$

$$y_n \in \{0,1\} \qquad \forall n \in N_C, \tag{9}$$

where

$$\text{MRP}_d(y) = \min \sum_{a \in A_I^d \cup A_C^d} \sum_{l \in L^d} c_l^d(a) z_l^d(a)$$

$$\text{s.t.} \sum_{a \in \delta^d_{\text{in}}(n)} \sum_{l \in L^d} z^d_l(a) \leq y_n q_n \qquad \forall n \in N_C, \tag{10}$$

$$\sum_{a \in \delta^d_{\text{in}}(n)} z^d_l(a) = \sum_{a \in \delta^d_{\text{out}}(n)} z^d_l(a) \qquad \forall n \in N^{dy}_F \setminus \{\mathcal{S}, \mathcal{T}\}, \quad \forall l \in L^d, \tag{11}$$

$$z^d_l(a) = m^d_l \qquad \forall l \in L^d, \quad a \in \delta^d_{\text{in}}(n^d_l), \tag{12}$$

$$\sum_{a \in \delta^d_{\text{in}}(\mathcal{T})} \sum_{l \in L^d} z^d_l(a) = \sum_{l \in L} m^d_l, \tag{13}$$

$$\sum_{a \in A^d_s} \sum_{l \in L^d} z^d_l(a) \leq g^d_s \qquad \forall s \in S^d, \tag{14}$$

$$\sum_{a \in A^d_I} \sum_{l \in L^d} z^d_l(a) \leq G^d, \tag{15}$$

$$z^d_l(a) \geq 0 \qquad \forall a \in A^{dy}_F, \quad \forall l \in L^d. \tag{16}$$

In the above formulation (RMLRP-2SRO) and (SMLRP-2SSO) optimize the worst/expected case cost, under Constraint (8). Constraint (8) guarantee that the opened facilities have enough combined capacity to handle all maintenance visits, so that the second stage problem has a feasible solution. $\text{MRP}_d(y)$ optimizes the maintenance routing as second stage decision. Constraints (10) guarantee that maintenance visits can only be assigned to opened facilities and no more than its capacity. Constraints (11) are the flow conservation constraints, while Constraints (12) and (13) guarantee that every maintenance visit is assigned to a facility. Constraints (14) and (15) are the end station and budget interchange capacity constraints.

5 Mixed Integer Formulation

The two-stage formulations of Section 4 are not practical in terms of computation. In this section we reformulate these problems into a mixed integer programming formulation. Then we provide an improved mixed integer programming formulation that is equivalent in terms of the optimal objective and optimal facility decisions, but that not with respect to the second-stage decisions. This improved formulation is very efficient to find optimal facility decisions, after which the optimal second-stage decisions can easily be computed if needed.

The first step to reformulate the two-stage models (RMLRP-2SRO) and (SMLRP-2SSO) to mixed integer models is to make one large maintenance routing flow graph. This graph contains all scenarios, instead of one graph for every scenario. The steps used in Section 3 are adapted to generate the graph $G_M = (N_M, A_M)$:

- For every scenario and line, we make a node. The set with all line nodes belonging to a scenario $d \in D$ is denoted as N^d_L.
- We have one source $\mathcal{S}$ that is shared for all scenarios. The source is connected with an arc to each node in $\bigcup_{d \in D} N^d_L$.

- We have an arc between every line where an interchange is possible (connected, same rolling stock type, same scenario), with as cost the interchange coordination cost.
- We make a node for every candidate facility, and each of these nodes is connected with an arc to the sink $\mathcal{T}$. The set of candidate facility nodes is denoted by N_C.
- For each node $n \in \bigcup_{d \in D} N_L^d$, we make an arc to each facility. The cost of this arc is the dead heading cost of the line to the facility. The cost of the arc is zero when dead heading is not necessary because the line associated with the node is connected to the facility.

The sets N_M^d and A_M^d contain all nodes or arcs that can be reached by flow from scenario $d \in D$.

The two-stage models (RMLRP-2SRO) and (SMLRP-2SSO) can be reformulated to the following mixed integer programming formulations:

$$\text{(RMLRP-MIP)} \quad \min \sum_{n \in N_C} c_n y_n + z_{\max}$$

$$\text{(SMLRP-MIP)} \quad \min \sum_{n \in N_C} c_n y_n + \sum_{d \in D} p_d \sum_{a \in A_I^d \cup A_C^d} \sum_{l \in L^d} c_l^d(a) z_l^d(a)$$

Subject to:

$$\sum_{a \in \delta_{\text{in}}^d(n)} \sum_{l \in L^d} z_l^d(a) \leq y_n q_n \qquad \forall n \in N_C, \quad \forall d \in D, \tag{17}$$

$$\sum_{a \in \delta_{\text{in}}^d(n)} z_l^d(a) = \sum_{a \in \delta_{\text{out}}(n)} z_l^d(a) \qquad \forall d \in D, \quad \forall n \in N_M^d \setminus \{\mathcal{S}, \mathcal{T}\}, \quad \forall l \in L^d, \tag{18}$$

$$z_l^d(a) = m_l^d \qquad \forall d \in D, \quad \forall l \in L^d, \quad \forall a \in \delta_{\text{in}}^d(n_l^d), \tag{19}$$

$$\sum_{d \in D} \sum_{a \in \delta_{\text{in}}^d(\mathcal{T})} \sum_{l \in L^d} z_l^d(a) = \sum_{d \in D} \sum_{l \in L^d} m_l^d, \tag{20}$$

$$\sum_{a \in A_s^d} \sum_{l \in L^d} z_l^d(a) \leq g_s^d \qquad \forall d \in D, \quad \forall s \in S^d, \tag{21}$$

$$\sum_{a \in A_I^d} \sum_{l \in L^d} z_l^d(a) \leq G^d \qquad \forall d \in D, \tag{22}$$

$$z_l^d(a) \geq 0 \qquad \forall d \in D, \quad \forall a \in A_M^d, \quad \forall l \in L^d, \tag{23}$$

$$y_n \in \{0, 1\} \qquad \forall n \in N_C, \tag{24}$$

with as additional Constraints for the RMLRP-MIP:

$$z_{\max} \geq \sum_{a \in A_I^d \cup A_F^d} \sum_{l \in L^d} c_l^d(a) z_l^d(a) \qquad \forall d \in D. \tag{25}$$

Constraints (17) to (23) are comparable to (10) to (16) from Section 4, however now defined for the graph $G_M = (N_M, A_M)$ and for all scenarios. Note that in the maintenance location

routing graph G_M, all arcs besides those going to the sink can only be traversed by flow from one scenario. Consequently, by the use of A_M^d and N_M^d in place of A_M and N_M, we removed many unnecessary constraints in Constraints (18) and (23). Constraints (25), which are only used for the RMLRP-MIP, guarantee that $z_{\max}$ equals the routing cost of the worst case scenario. The SMLRP-2SSO and SMLRP-MIP are equivalent. However, for the RMLRP-2SRO and RMLRP-MIP only the opened and closed facilities y^* and the objective value are equivalent. The routings for the non binding scenarios, can be different from those of the RMLRP-2SRO, because only the binding scenario is restricted by $z_{\max}$. However, if necessary, the equivalent routing for the (non binding) scenarios can be found in polynomial time by solving $\text{MRP}_d(y^*)$ $\forall d \in D$ for the optimal RMLRP-MIP solution y^*.

5.1 Improved Mixed Integer Model

The mixed integer formulations can be improved by using two observations. The first observation is that we do not need to distinguish the scenario for each flow. Without the index d, we can still determine to which scenario a flow belongs based on the incoming arcs of the facility nodes as they always belong to only one scenario. The second observation is that the index l can be dropped as well. Without the index l, we cannot be determine from which line a flow originates and the route it has taken. However, the incoming flow per facility will be the same, resulting in the same facilities opened and the same objective value. Removing these variables results in a much more efficient formulation. However, to find the routes and the originating lines for a scenario $d \in D$, we have to solve the maintenance routing problem $\text{MRP}_d(y^*)$ from Section 3, where y^* is the optimal R/SMLRP-IMIP solution. This can be done in polynomial time.

Furthermore, Constraints (20) can be omitted, because without the l indices, all flow is already restricted by Constraints (19). Additionally, in Constraints (17) q_n can be a lot higher than the total number of maintenance visits ($\sum_{l \in L^d} m_l^d$) for a scenario $d \in D$. Tightening Constraints (17), by replacing q_n by $\hat{q_n^d} = \min(q_n, \sum_{l \in L^d} m_l^d)$ for every scenario d can improve the initial LP bound. When the d index of a set is dropped, we take the union of the sets for all scenarios. This gives the following improved mixed integer programming (IMIP) formulations:

$$\text{(RMLRP-IMIP)} \quad \min \sum_{n \in N_C} c_n y_n + z_{\max}$$

$$\text{(SMLRP-IMIP)} \quad \min \sum_{n \in N_C} c_n y_n + \sum_{d \in D} p_d \sum_{a \in A_I^d \cup A_C^d} c(a) z(a)$$

Subject to:

$$\sum_{a \in \delta^d_{\text{in}}(n)} z(a) \leq y_n \hat{q_n} \qquad \forall n \in N_C, \quad \forall d \in D, \tag{26}$$

$$\sum_{a \in \delta_{\text{in}}(n)} z(a) = \sum_{a \in \delta_{\text{out}}(n)} z(a) \qquad \forall n \in N_M \setminus \{\mathcal{S}, \mathcal{T}\}, \tag{27}$$

$$z(a) = m^d_l \qquad \forall d \in D, \quad \forall l \in L^d, \quad \forall a \in \delta^d_{\text{in}}(n^d_l), \tag{28}$$

$$\sum_{a \in A^d_s} z(a) \leq g^d_s \qquad \forall d \in D, \quad \forall s \in S^d, \tag{29}$$

$$\sum_{a \in A^d_I} z(a) \leq G^d \qquad \forall d \in D, \tag{30}$$

$$z(a) \geq 0 \qquad \forall a \in A_M, \tag{31}$$

$$y_n \in \{0, 1\} \qquad \forall n \in N_C, \tag{32}$$

with as additional Constraints for the (RLMRP-IMIP):

$$z_{\max} \geq \sum_{a \in A^d_I \cup A^d_C} c(a) z(a) \qquad \forall d \in D. \tag{33}$$

Theorem 1. *The R/SMLRP-IMIP has an identical optimal cost and facility decision y^* as the R/SMLRP-MIP.*

Proof. For every $a \in \delta_{\text{out}}(\mathcal{S})$, $z^d_l(a) \geq 0$ for only one scenario d and line l, and for all other scenarios and lines $z^d_l(a) = 0$. This is the case because of Constraint (19) and (20) which guarantee that every line node receives only m^d_l flow for one l and d and 0 for all other lines and scenarios. These line nodes are only connected to the line nodes of the same scenario and the facility nodes. Consequently, the line nodes can only receive flow of one scenario type, and the scenario of the incoming flow of the facilities, is equal to the scenario to which the line node from which the incoming facility flow originates from belongs to. Furthermore, dropping the indices will not influence the amount of flow for every arc, as $z(a)$ will automatically be equal to $\sum_{d \in D} \sum_{l \in L^d} z^d_l(a)$. Consequently, the same facilities will be opened and the objective value will be the same. □

Theorem 2. *The R/SMLRP-IMIP reduces the number of variables by $\Theta(\sum_{d \in D} \left(|L^d|^2 |N_C| + |L^d||A^d_I|\right))$ compared to the R/SMLRP-MIP.*

Proof. Note that $|A_F| = \sum_{d \in D}(|L^d| + |L^d||N_C| + |A^d_I|) + |N_C|$ and $|A^d_F| = |L^d| + |L^d||N_C| + |A^d_I| + |N_C|$.

We subtract the number of variables in the IMIP from the number of MIP variables.

The number of removed variables is:

$$
\begin{aligned}
&\sum_{d\in D}\left(|L^d||A_F^d|\right) - |A_F| \\
&= \sum_{d\in D}\left(|L^d|(|L^d| + |L^d||N_C| + |A_I^d| + |N_C|)\right) - \sum_{d\in D}(|L^d| + |L^d||N_C| + |A_I^d|) - |N_C| \\
&= \sum_{d\in D}\left(|L^d|^2 + |L^d|^2|N_C| + |L^d||A_I^d| + |L^d||N_C|\right) - \sum_{d\in D}(|L^d| + |L^d||N_C| + |A_I^d|) - |N_C| \\
&= \sum_{d\in D}\left(|L^d|^2 + |L^d|^2|N_C| + |L^d||A_I^d| + |L^d||N_C| - |L^d| - |L^d||N_C| - |A_I^d|\right) - |N_C| \\
&= \sum_{d\in D}\left(|L^d|^2 + |L^d|^2|N_C| + |L^d||A_I^d| - |L^d| - |A_I^d|\right) - |N_C|
\end{aligned}
$$

Consequently, the number of variables removed is of order $\Theta(\sum_{d\in D}\left(|L^d|^2|N_C| + |L^d||A_I^d|\right))$. □

6 Scenario Addition Method and Benders Decomposition

We design a row-and-column generation algorithm (similar to Zeng and Zhao, 2013), also called column-and-constraint generation, for the RMLRP. Our method, called Scenario Addition (SA), solves the IMIP with a subset of scenarios and adds an additional scenario each iteration. The algorithm reaches optimality in a finite number of iterations. The SA algorithm uses the fact that for the RMLRP only the facility solution y^* and the worst case scenario are relevant.

Furthermore, we apply Benders decomposition (Benders, 1962), also known as the L-shaped decomposition method in the stochastic programming literature (van Slyke and Wets, 1969) to both the RMLRP as the SMLRP. Benders decomposition is a well know procedure where an optimization problem is solved by defining it in terms of a master and a slave problem. It effectiveness relies on the possibility of adding good optimality cuts to the master problem, ruling out many of the trial values for the master problem. We accelerate the Benders decomposition by making use of several different accelerating heuristics.

6.1 Improved Maintenance Routing Model

It is necessary to solve the second stage problem (MRP) from Section 4 many times for both SA and Benders decomposition. The MRP can be improved by using Theorem 1 from Section 5. This theorem allows us to drop the l indices. Although the MRP is a polynomial problem, it still gives a very substantial reduction in solution time. For larger instances (100 facilities, 64 scenarios) the difference in solution time is from multiple seconds per scenario, to solving all scenarios in approximately a second. Furthermore, similar to Section 5, the capacity q_n, can be replaced by $\hat{q_n^d}$.

The second stage problem becomes:

$$\text{IMRP}_d(y) = \min \sum_{a \in A_I^d \cup A_C^d} c(a) z(a)$$

Subject to:

$$\sum_{a \in \delta_{\text{in}}(n)} z(a) \leq y_n \hat{q_n^d} \qquad \forall n \in N_C, \qquad (\mu)$$

$$\sum_{a \in \delta_{\text{in}}(n)} z(a) = \sum_{a \in \delta_{\text{out}}(n)} z(a) \qquad \forall n \in N_F^{dy} \setminus \{\mathcal{S}, \mathcal{T}\}, \qquad (\nu)$$

$$z(a) = m_l^d \qquad \forall l \in L^d, \quad a \in \delta_{\text{in}}^d(n_l^d), \qquad (\pi)$$

$$\sum_{a \in A_s^d} z(a) \leq g_s^d \qquad \forall s \in S^d, \qquad (\phi)$$

$$\sum_{a \in A_I^d} z(a) \leq G^d, \qquad (\omega)$$

$$z(a) \geq 0 \qquad \forall a \in A_F^{dy}.$$

The Greek symbols next to the constraints are the corresponding dual variables, which we need for the Benders decomposition algorithm described in Section 6.3.

6.2 Scenario Addition Method

For the RMLRP, only the opened facilities and the worst case scenario are relevant. This method makes use of that fact, and can consequently not be used for the SMLRP where all scenarios contribute to the objective function.

The method works as follows: Set the iteration counter i at 0 and let D^i denote the scenario set belonging to iteration i. The set D^0 contains one randomly chosen scenario $d \in D$. Because the scenarios differ in the total number of maintenance visits, the following feasibility constraint is added to the RMLRP-IMIP:

$$\sum_{n \in N_C} y_n \hat{q}_n \geq \max_{d \in D} \sum_{l \in L^d} m_l^d. \qquad (34)$$

The scenario addition algorithm consists of the following steps:

1. Compute the solution to the RMLRP-IMIP with D replaced by D^i, and add Constraint (34) to the formulation. Denote the optimal objective of this problem as LB^i and the solution of iteration i as y^i.

2. Calculate for each scenario $d \in D$, $\text{IMRP}_d(y^i)$ separately. Set $\text{UB}^i = c^T y^i + \max_{d \in D} \text{IMRP}_d(y^i)$.

3. If $\text{UB}^i = \text{LB}^i$ stop and return y^i as the optimal solution and UB as the optimal objective value. Otherwise the algorithm proceeds to the next step.

4. The scenario w^i is the scenario with the worst $\text{IMRP}_d(y^i)$ from the set $D \setminus D^i$. The set $D^{i+1} = \{w^i\} \cup D^i$. Update $i = i + 1$ and go back to Step 1.

These steps give a very diverse set of scenarios as it always adds the worst case scenario. Because of this, the algorithm generally converges quickly, with only a small number of scenarios in D^i.

6.3 Benders Decomposition

Benders decomposition can be used for both the RMLRP as the SMLRP. The algorithm iteratively solves a master and a slave problem. The master problem gives a lower approximation of the optimal objective value and provides a lower bound, LB. We define y^i as the optimal solution of the master problem for iteration i. The slave problem is equal to improved second stage problem described in Section 6.1. The slave problem calculates the actual objective value of the R/SMLRP given facility set y^i, and provides an upper bound UB. When the lower and upper bound are sufficiently close to each other, we conclude optimality. Otherwise we use information from the duals of our slave problem to add additional optimality cuts to the master problem. The optimality cuts serve to improve the estimation of the optimal value of the master problem. It increases the lower bound, changes the solution y^i, and consequently can also increase the upper bound.

The lower bound LB is set initially at $-\infty$ and the upper bound UB at ∞. The iteration counter is set at $i = 0$, and we let $\hat{y}$ be the incumbent solution. The Benders decomposition method consists of the following steps:

1. Solve the master problem. The master problem is the first stage problem from Section 4 with optimality cuts representing input from the second stage.

$$\text{LB} = \min_{y,\theta} \sum_{n \in N_C} c_n y_n + \theta$$

Subject to:

$$\sum_{n \in N_C} y_n q_n \geq \max_{d \in D} \sum_{l \in L^d} m_l^d, \tag{35}$$

$$y_n \in \{0, 1\} \qquad \forall n \in N_C, \tag{36}$$

$$\theta \geq \sum_{n \in N_C} a_{kn} y_n + b_k \qquad k = 1, \ldots, i, \tag{37}$$

$$\theta \geq 0. \tag{38}$$

The variable θ represent the maintenance routing costs, and Constraints (37) approximate the constraints $\theta \geq \max_{d \in D} \text{IMRP}_d(y)$ for the RMLRP and $\theta \geq \sum_{d \in D} p_d \text{IMRP}_d(y)$ for the SMLRP. Optimality Constraints (37) are only added to the master problem from iteration 1, and will be ignored the first time that it is solved. The coefficients $a_{i+1,n}$ and b_{i+1} are

determined from the duals from the $\text{IMRP}_d(y^i)$ in such a way that $\sum_{n\in N_C} a_{i+1,n}y^i_n + b_{i+1} = \max_{d\in D} \text{IMRP}_d(y^i)$, and $\sum_{d\in D} p_d \text{IMRP}_d(y^i)$, for the RMLRP and SMLRP respectively. In Step 4, it is explained in detail how the variables $a_{i+1,n}$ and b_{i+1} are generated.

2. Calculate the objective value: $\text{SMLRP}(y^i) = c^T y^i + \sum_{d\in D} p_d \text{IMRP}(y^i, d)$, or $\text{RMLRP}(y^i) = c^T y^i + \max_{d\in D} \text{IMRP}(y^i, d)$. If $\text{SMLRP}(y^i)$ or $\text{RMLRP}(y^i)$ is smaller than the upper bound, we update the upper bound with the new objective value and set the incumbent solution at $\hat{y} = y^i$.

3. If $\text{UB} - \text{LB} < \delta$, where $\delta > 0$ is a pre-specified tolerance, the algorithm stops and returns $\hat{y}$ as the optimal solution and UB as the optimal objective value. Otherwise the algorithm proceeds to Step 4.

4. Let $\mu^d_{in}, \nu^d_{inl}, \pi^d_{il}, \phi^d_{is}$ and ω^d_i be the dual variables for $\text{IMRP}(y^i, d)$ (see Section 6.1). The cut coefficients for the RMLRP are:

$$a_{i+1,n} = \mu^{\text{dmax}}_{in} \hat{q^d_n} \quad \forall n \in N_C$$

and

$$b_{i+1} = \sum_{l\in L} \pi^d_{il} m^d_l + \sum_{s\in S^d} \phi^{\text{dmax}}_{is} g^{\text{dmax}}_s + \omega^{\text{dmax}}_i G^{\text{dmax}},$$

where $\text{dmax} = \arg\max_{d\in D} \text{MRP}(y^i, d)$.

For the SMLRP the cut coefficients are:

$$a_{i+1,n} = \sum_{d\in D} p_d \mu^d_{in} \hat{q^d_n} \quad \forall n \in N_C$$

and

$$b_{i+1} = \sum_{d\in D} p_d \left(\sum_{l\in L} \pi^d_{il} m^d_l + \sum_{s\in S^d} \phi^d_{is} g^d_s + \omega^d_i G^d \right).$$

Update $i = i + 1$ and go back to Step 1.

6.3.1 Acceleration Techniques

Although Benders decomposition terminates in a finite number of iterations, it still can take an exponential number of iterations. The following accelerating techniques improve the convergence behaviour of the standard Benders decomposition.

Upper Bound Heuristics. The R/SMLRP-IMIP from Section 5.1 with a subset of the scenarios can be used to get a good UB. Note that not all scenarios have the same number of maintenance visits, consequently to guarantee feasibility we need to add Constraint (34), from Section 6.2. Let y^0, be the optimal solution of the R/SMLRP-IMIP with a subset of the scenarios and feasibility constraints (34). Then the upper bound can be calculated by solving Step 2 of

the Benders decomposition with y^0 as input: UB = SMLRP(y^0) = $c^T y^0 + \sum_{d \in D} p_d \text{IMRP}_d(y^0)$ or RMLRP(y^0) = $c^T y^0 + \max_{d \in D} \text{IMRP}_d(y^0)$. Use Step 3 and 4 to generate and add the cut coefficients. Update $i = i + 1$ and go back to Step 1.

With a good UB and starting y^0, good cuts are directly added to the master problem, which avoids the algorithm from exploring inferior parts of the feasible region. For the SMLRP it makes sense to order the scenarios at decreasing probabilities and to add the scenarios with the highest probability to the subset. Because the probabilities do not sum up to 1 anymore, it is best to renormalise them such that the sum of the scenarios in the subset becomes 1. The more scenarios the subset uses, the better the UB becomes, however the more time it costs to solve the IMIP.

Trust Regions. A trust region helps the algorithm avoid oscillating wildly from one region of the feasible set to another in the early iterations, and consequently accelerates convergence. For the same reasons as Santoso et al. (2005), we use a trust region based on the Hamming distance of the current master problem from the previous. Let Y^i be $\{j : y_j^i = 1\}$, then the trust region constraint for iteration $i + 1$ is:

$$\sum_{j \in Y^i} (1 - y_j) + \sum_{j \notin Y^i} y_j \leq \Delta_H^{i+1}, \tag{39}$$

where $\Delta_H^{i+1} \leq |N_C|$ is the trust region size at iteration $i+1$. Δ_H^{i+1} can be constant or dependent on the iteration. Because convergence of the algorithm cannot be ensured with a trust region, it is best to only add the trust region in the initial iterations, and drop it when the iterations have stabilized.

We also introduce a new trust region, where we use the incumbent solution $\hat{y}$ instead of y^i. Like before, we solve the master problem, however now we also solve it a second time with an additional constraint:

$$\sum_{n \in N_C} \hat{y}_n - \sum_{n \in N_C} y_n \leq \Delta_I^{i+1}, \tag{40}$$

where $\Delta_I^{i+1} \leq |N_C|$ can be constant or dependent on the iteration. This constraint can be useful, because in the initial iterations of the MLRP, far too few facilities are opened. The master problem combined with Constraint (40) gives solutions which are closer to the optimal number of opened facilities, while also using essential information from the master problem. It gives an additional heuristic solution, for which additional optimality cuts can be added or which can replace the regular optimality cuts when the solution value calculated in Step 2 is better. When the regular master problem solution already satisfies the constraint, the two solutions will be the same, and it is not necessary to solve the master problem twice. It is best to combine this trust region, with upper bound heuristics, so that the initial incumbent solution is already decent. A benefit of this trust region is that convergence is guaranteed so Constraint (40) does not have to be dropped after some number of iterations.

Knapsack Inequalities. We use the knapsack inequalities described in Santoso et al. (2005). With a good upper bound, knapsack inequalities can accelerate the Benders decomposition. Let $\theta \geq \sum_{n \in N_C} a_{in} y_n + b_i$ be the optimality cut from the ith iteration. Furthermore, we know that the current best known upper bound, $\text{UB} \geq \sum_{n \in N_C} c_n y_n + \theta$. These two facts together give the following inequality for the master problem in iteration $i+1$:

$$\sum_{n \in N_c} (\lfloor c_n + a_{in} \rfloor) y_n \leq \lfloor \text{UB} - b_i \rfloor.$$

Cut Disaggregation. Cut disaggregation can only be used for the SMLRP. Instead of one cut per iteration that approximate the average second-stage value, D cuts are added to approximate the individual second-stage value corresponding to each of the scenarios. Cut disaggregation (also called multi cut framework) can provide a better estimation of the average of the second-stage value, and thereby improve performance. Birge and Louveaux (1988) show that such a framework can greatly increase convergence. The trade-off lies in the fewer iterations of the Benders decomposition and the larger master problem. The master problem now becomes:

$$\text{LB}= \min_{y,\theta} \sum_{n \in N_C} c_n y_n + \sum_{d \in D} p_d \theta_d$$

Subject to:

$$\sum_{n \in N_C} y_n \hat{q}_n \geq \sum_{l \in L^d} m_l^d \qquad \forall d \in D, \tag{41}$$

$$y_n \in \{0,1\} \qquad \forall n \in N_C, \tag{42}$$

$$\theta_d \geq \sum_{n \in N_C} a_{kn}^d y_n + b_k^d \qquad k = 1, \ldots, i, \quad d \in D, \tag{43}$$

$$\theta_d \geq 0 \qquad \forall d \in D, \tag{44}$$

and the coefficients can be calculated with:

$$a_{i+1,n}^d = \mu_{in}^d \hat{q_n^d} \quad \forall n \in N_F, \quad \forall d \in D$$

and

$$b_{i+1}^d = \sum_{l \in L} \pi_{il}^d m_l^d + \sum_{s \in S^d} \phi_{is}^d g_s^d + \omega_i^d G^d \quad \forall d \in D.$$

Cut Strengthening. We also tested cut strengthening as proposed by Magnanti and Wong (1981). When ($\grave{a}$,$\grave{b}$) and ($\acute{a}$,$\acute{b}$) are both optimality cut coefficients at y^i corresponding to distinct dual optimal solutions, then $\grave{a}^T y^i + \grave{b} = \acute{a}^T y^i + \acute{b}$. Suppose that y^* is the optimal solution then cut ($\grave{a}$,$\grave{b}$) dominates cut ($\acute{a}, \acute{b}$) if $\grave{a}^T y^* + \grave{b} > \acute{a}^T y^* + \acute{b}$. Clearly cut ($\grave{a}$,$\grave{b}$) is preferable in iteration i since it will typically lead to better lower bounds and expedite convergence. Finding stronger cuts can be done, however it takes time. In our case, we are seldom able to find iterations were the algorithm can find better cuts than those of the original Benders decomposition. We expect

that this is because or our improvements from Section 6.1, which removed a lot of degeneracy. Consequently, for our problem, cut strengthening increases the solution time, as it takes time to try to strengthen the cuts, while we seldom benefit from it.

7 Computational Experiments

In this section, we report computational experiments on randomly generated instances to test the computational performance of our developed algorithms. We are particularly interested in the size of instances that can be solved by the IMIP, SA and Benders decomposition. Although we generate instances randomly, the fixed and random parameters are based on those found in practice to create reasonable instances. All experiments are programmed in Java with the CPLEX library version 12.6.3, and run on a laptop with an Intel Core i7-4710MQ Quad Core 2.5 GHz processor with 8GB of RAM. All mentioned solution times include the time necessary to build the model and for all settings (preprocessing, branching, accuracy tolerance etcetera) CPLEX standard settings are used.

7.1 Test Bed Generation

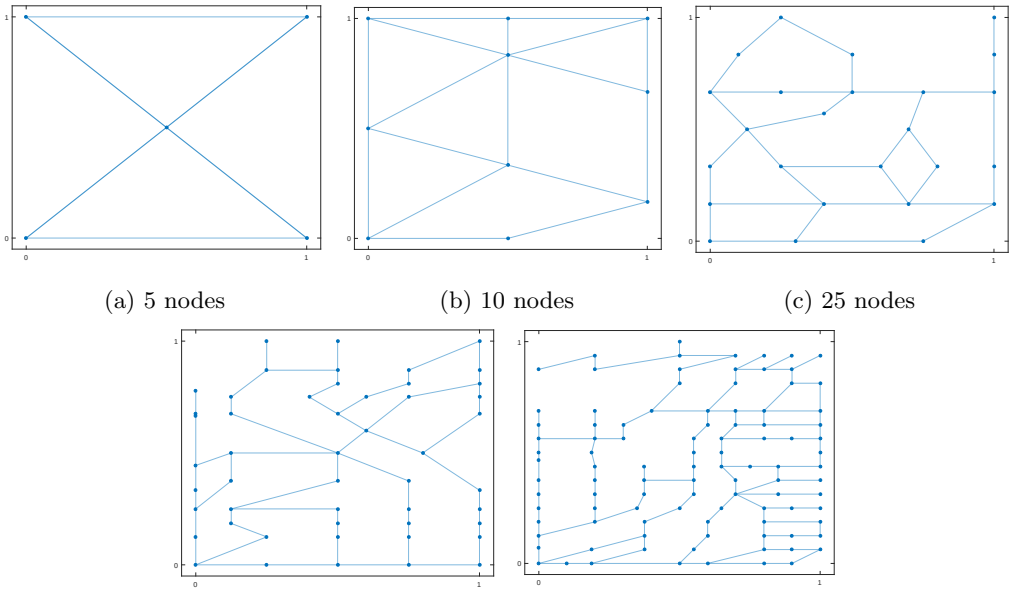
We generate instances with 5, 10, 25, 50 and 100 candidate facilities. We start with 15 instances with only 1 scenario, for each number of candidate facilities. These 15 instances consist of 5 instances of each of the three sizes which will be described in Section 7.1.1. In each next set we double the number of scenarios, until the instances within the set can not be solved anymore.

7.1.1 Physical Railway Network and Basic Line Planning Generation

We create random instances by first generating graphs of physical railway networks in the cartesian plane. We do this by manually generating five archetypical graphs on a one by one plane with respectively 5, 10, 25, 50, and 100 nodes as shown in Figure 3. Each node is an end station and candidate facility location, while each edge is a set of rails between end stations. For each archetypical graph, we also manually designed "basic line planning" scenarios. A basic line plan consists only of origin-destination pairs and the route between them. We also determine for each basic line whether the corresponding rolling stock type will be regional or intercity. Note however, that there may be several regional and intercity rolling stock types, and that the actual rolling stock type will be assigned in Section 7.1.2.

We create physical railway networks by perturbing and scaling these archetypical networks. We perturb these archetypical graphs by relocating each node uniformly at random within a square around that node. The square is centered around the original node except at the boundaries of the 1×1 plane where the square is relocated to fit within the 1×1 plane. The dimension of this square decreases with the number of nodes in the network. Next the obtained graph is scaled to either a (1) 200×400 km, (2) 750×1000 km, or (3) 3000×3000 km rectangle to model

different railway network sizes. The distances between nodes in this graph is then determined by multiplying the Euclidian distance by a number drawn uniformly between 1.0 and 1.2. Note that at this point the length in kilometers of each basic line is fixed. A graph thus obtained is called $G_P = (N_P, E_P)$.



(a) 5 nodes (b) 10 nodes (c) 25 nodes

(d) 50 nodes (e) 100 nodes

Figure 3: The five archetypical graphs.

7.1.2 Line Plan Generation

Recall that a train line consist of a rolling stock type, the maintenance frequency per year and the deadheading cost to each facility. Before we can generate the line plan, we first need to generate the different types of rolling stock that we can use. We differentiate between regional and intercity trains (the pre-defined type in the basic line planning), and generate the number of different rolling stock types for each material type randomly. This number is generated uniformly between 1 and the number of edges $|N_P|$ divided by 8. Per rolling stock type dependent if its regional or intercity, all necessary data to calculate the dead heading cost per kilometer (fuel, driver, unavailability cost etc) is generated randomly. Furthermore, to be able to calculate the number of maintenance visits for a line later on, we need to know the average number of yearly maintenance visits per rolling stock type, this number is generated uniformly between 1 and 12.

Then we generate line planning scenarios until we have reached the fixed number of scenarios for the instance. We start with generating a line planning, by taking uniformly at random a line planning possibility from the fixed basic line planning set. For each line of the chosen basic line planning set, we generate the rolling stock type and maintenance visit frequency. The rolling

stock type is generated based on the pre-defined material type. The rolling stock type is the same as that of a connecting line of the same basic type with a probability of 0.5 for each line. When there are no defined lines yet, or when the rolling stock type is not going to be the same as an already defined connecting line, we choose uniformly at random from the set with rolling stock types for the basic rolling stock type of the line.

The yearly maintenance frequency is the average number of train units on that line times the yearly maintenance frequency of the rolling stock type used for that line. An estimation for the average number of train units for a train line is made based on its length in kilometers, the average speed of the trains and the hourly train frequency of the line, which is generated uniformly at random between 1 and 8. The deadheading cost for a line to a facility is the deadheading cost per kilometer of the rolling stock type times the number of kilometers for the shortest path in G_P from one of the end stations nodes to the facility node. For example when one of the end stations is the candidate facility, the shortest path is 0 kilometers and the cost is 0.

7.1.3 Interchange Capacities, Budgets and Candidate Facilities

All interchange coordination cost between lines are set at 10 euro. The interchange capacity of an end station, g_s^d, are based on the sum of maintenance visits of all lines with end station s in scenario d. With a probability of 5%, no interchanges are possible, and with a probability of 20% the capacity is uniformly distributed between 0 and 1 times the sum of maintenance visits. With a probability of 65% it is uniformly distributed between 1 and 3 times the sum of maintenance visits, only a restriction when there are many interchanges originating from non connecting lines, and with a probability of 10%, there is no restriction at all. The interchange budget G^d is with a probability of 25% between 0 and 1 times the total yearly maintenance visit frequency for scenario d, with a probability of 65% between one and three times the total yearly maintenance visit frequency and with a probability of 10% it is infinite.

The fixed facility costs are generated uniformly at random between 1/5th and 5 times the estimated average fixed facility cost for the Dutch Railways. The maintenance facility capacities are with a probability of 0.1 uniformly distributed between 0 and 0.5 times the maximum total yearly number of maintenance visits over all scenarios, with 0.4 between 0.5 and 1 and with a probability of 0.5 there is no capacity restriction.

For the SMLRP, we also have to generate the probabilities p_d. These probabilities are uniformly generated between 0 and 1, followed by scaling the probabilities such that they sum up to 1.

7.2 IMIP Experiments

We used the instances with 5, 10, 25, 50 and 100 candidate facilities from the randomly generated instance sets. For each number of facilities, we start with the set with just one scenario. When

more than 80% (13 or more out of 15) of the instances can be solved within an hour, we continue to the next set where the number of scenarios is doubled. When an instance can not be solved within an hour, a fail is registered and a solution time of 3600 seconds is used for the average time calculations. The average time in seconds for each set for the R/SMLRP can be found in Table 1. The top row shows the number of candidate facilities per set and the first column the number of scenarios.

	RMLRP - IMIP					SMLRP - IMIP				
Scenarios	**5**	**10**	**25**	**50**	**100**	**5**	**10**	**25**	**50**	**100**
1	< 0.1	< 0.1	0.1	0.7	13.4	< 0.1	< 0.1	0.1	1.1	19.4
2	< 0.1	< 0.1	0.2	1.8	103.4	< 0.1	< 0.1	0.2	1.7	110
4	< 0.1	0.1	0.6	9.1	175.5	< 0.1	< 0.1	0.3	8.6	94.2
8	0.1	0.1	2.2	35.2	874.3	< 0.1	0.1	0.6	20.6	202.6
16	0.2	0.2	2.8	67.6	fails	0.1	0.2	2.5	33.2	956.6
32	0.2	0.3	5.3	231.0		0.1	0.3	5.7	112.5	fails
64	0.3	0.6	13.5	715.4		0.2	0.5	10.4	296.7	
128	0.5	1.7	71.4	1259.3		0.4	1.4	37.6	785.8	
256	1.6	4.2	140.1	fails		1.1	3.0	112.9	fails	
512	5.0	15.2	452			2.9	11.8	507.4		
1024	16.3	44.9	1259.8			10.1	36.0	1076.8		
2048	54.7	178.5	fails			31.1	154.8	fails		
4096	214.4	531.2				126.8	653.4			
8192	733.2	1978.1				556.7	fails			
16384	fails	fails				2428.3				
32768						fails				

Table 1: Solution time in seconds for the RMLRP and SMLRP, while varying the number of scenarios and facilities.

For up to 50 facilities the IMIP solved with CPLEX works well. Note that these instances are already large. An instance with 25 facilities and 1024 scenarios has for the RMLRP on average 36 k (35765.5) nodes and a million (1007822.6) edges in the flow graph. Furthermore, the CPLEX model has a million (1007848.6) columns and 100k (100123.1) rows. For 100 candidate facilities, we can only solve the instances with a limited number of scenarios within an hour. For these instances, we extended the solution time to 48 hours (2880 minutes), to explore the computationally feasible boundary. Because of time considerations, we limit the instances of the sets to the first 6 instances, 2 instances from each size. These results are depicted in Table 2, where RAM is used for instances where an out of memory error occurred.

Instances up to 64 scenarios can generally be solved. Furthermore, the SMLRP-IMIP seems

	RMLRP-IMIP				SMLRP-IMIP			
Scenarios	**Min**	**Average**	**Max**	**Failed**	**Min**	**Average**	**Max**	**Failed**
16	17.4	105.6	252.6	0	0.4	12.1	56.2	0
32	22.8	429.6	791.4	0	1.5	72.3	331.5	0
64	123.0	631.8	2541.0	0	4.6	209.2	892.7	0
128	168.7	2337.7	> 2880.0	4	6.6	595.9	> 2880.0	2
256	RAM	RAM	RAM	6	18.5	774.5	> 2880.0	1
512	RAM	RAM	RAM	6	RAM	RAM	RAM	6

Table 2: Minimum, average and maximum solution time in minutes for instances with 100 facilities and the number of instances which were not solved within 2880 minutes.

to be computationally faster than the RMLRP-IMIP. A possible explanation for this are Constraints (33) from Section 5.1, which are only used for the RMLRP-IMIP. These constraints create degeneracy, because with them only one scenario contributes to the transportation cost, giving many solutions with the same objective value.

Furthermore, we noticed that the IMIP (solved with CPLEX standard branch and cut algorithm) uses many cuts and few nodes in the branch and bound tree. Some of the instances can be solved without branching. For example, the average number of nodes for the SMLRP with 50 candidate facilities is 72, while it uses 4226 cuts on average. The number of nodes increases with the number of candidate facilities, while the number of cuts increases by both the number of facilities and scenarios. CPLEX mainly uses flow and MIP rounding cuts, followed by implied bound cuts, and to a lesser extend clique, Gomory, zero-half and lift and project cuts.

7.3 Scenario Addition

We compared the scenario addition algorithm with the IMIP. We used the instances with 25, 50 and 100 candidate facilities from the randomly generated instance set. The instances with 5 and 10 facilities are excluded because the IMIP can already solve instances up to 8192 scenarios within an hour. Again more than 80% (13 or more out of 15) of the instances have to be solved within an hour to go to the next set. The average time in seconds for each set for the RMLRP-IMIP and SA can be found in Table 3; the top row shows the number of candidate facilities per set and the first column the number of scenarios.

For the instances with 25 and 50 candidate facilities, it can be seen in Table 3 that the SA algorithm can solve instances with more scenarios within the hour. For the instances with 25 candidate facilities, we stopped the test after 32384 scenarios, although they still can easily be solved within the hour. There is no difference between SA and IMIP for the instances with 100 facilities. This can be explained by the fact that we can only solve instances with a limited number of scenarios for instances with 100 facilities. Consequently, a SA iteration with fewer

	IMIP			SA		
Scenarios	**25**	**50**	**100**	**25**	**50**	**100**
1	0.1	0.7	13.4	0.1	0.7	13.5
2	0.2	1.8	103.4	0.3	2.4	129.1
4	0.6	9.1	175.5	1.2	14.1	220.9
8	2.2	35.2	874.3	2.7	42.2	719.7
16	2.8	67.6	fails	4.4	76.4	fails
32	5.3	231.0		5.1	328.7	
64	13.5	715.4		5.3	357.4	
128	71.4	1259.3		8.4	267.9	
256	140.1	fails		11.9	485.3	
512	452			12.8	528.9	
1024	1259.8			19.1	851.9	
2048	fails			40.5	503.6	
4096				54.9	1095.0	
8192				75.3	1073.2	
16384				144.3	fails	
32768				288.8		

Table 3: Solution time in seconds for the IMIP and SA, while varying the number of scenarios and facilities.

scenarios added already takes a considerable amount of time.

When we extend the time from 1 hour to 48 hours (we limit the sets to the first 6 instances, as explained in Section 7.2), the SA algorithm performs better in both the best and average case, but its worst case performance is as expected worse. We show these results in Table 4.

The number of iterations (is equal to the number of added scenarios) that the SA algorithm uses and the time in seconds is shown in Table 5. The number of iterations initially grows when the number of scenarios increases, to become stable after some point. The maximum number of iterations is really low compared to the total number of scenarios $|D|$ of the instances. The maximum number of scenarios for 25 facilities is 32384 while for the worst case, the scenario set only contains 20 scenarios. When we increase the number of candidate facilities to 50, the maximum number of scenarios is 8192, while the maximum number of iterations is 24.

7.4 Benders Decomposition

We use the instances with 25 candidate facilities and start with 128 scenarios. The upper bound heuristic always uses a set with 64 scenarios, with an average solution time around 10 seconds. The δ (absolute gap) is set at 0.0001. Just as for the IMIP experiments, we continue to the

	RMLRP-IMIP				SA			
Scenarios	**Min**	**Average**	**Max**	**Failed**	**Min**	**Average**	**Max**	**Failed**
16	17.4	105.6	252.6	0	1.2	74.4	198.0	0
32	22.8	429.6	791.4	0	5.4	560.4	2247.0	0
64	123.0	631.8	2541.0	0	8.4	511.2	> 2880.0	1
128	168.7	2337.7	> 2880.0	4	2.4	807.0	> 2880.0	2
256	RAM	RAM	RAM	6	66.0	1103.7	> 2880.0	2

Table 4: Minimum, average and maximum solution time in minutes for instances with 100 facilities and the number of instances which were not solved within 2880 minutes.

	25		50		100	
Scenarios	**Iterations**	**Time**	**Iterations**	**Time**	**Iterations**	**Time**
1	1.0	0.1	1.0	0.7	1	13.5
2	1.9	0.3	1.9	2.4	2	129.1
4	2.9	1.2	3.5	14.1	3.4	220.9
8	4.6	2.7	5.3	42.2	4.9	719.7
16	5.9	4.4	6.9	76.4		
32	7.3	5.1	9.3	328.7		
64	7.8	5.3	11.7	357.4		
128	7.1	8.4	11.2	267.9		
256	7.5	11.9	11.8	485.3		
512	7.9	12.8	10.9	528.9		
1024	7.9	19.1	10.9	851.9		
2048	8.1	40.5	11.6	503.6		
4096	8.3	54.9	12.5	1095.0		
8192	7.0	75.3	12.1	1073.2		
16384	6.7	144.3				
32768	6.7	288.8				
max	**21**	**983.1**	**24**	$>$ **3600**	**8**	$>$ **3600**

Table 5: The average and maximum number of iterations needed for the scenario addition method

next set when more than 80% (13 or more out of 15) of the instances are solved within an hour. Benders without any accelerating strategies can not solve any of the instances within the hour. All other accelerating techniques except for cut disaggregation (CD), are insufficient to speed-up the convergence such that it can solve the set of instances with 128 scenarios. CD seem to be necessary to have any potential success with Benders decomposition. We write down HTR, when

we used a trust region based on the Hamming distance. We tested different trust regions, and choose for a trust region where the Hamming distance is 1, for the first 5 iterations, 2 for iteration 6 to 10, and 3 up to iteration 25. Furthermore, we tested two versions of the incumbent trust region heuristic. In the first version, ITR A, we always add the optimality cuts from the best of the two solutions. In the second version, ITR B, we always add the regular optimality cuts and we add the additional cuts too when its solution is better than the regular master problem solution. We abbreviate the knapsack inequalities to KI.

Figure 6 depicts our results sorted in decreasing computational time (in seconds) for the SMLRP. We removed the combinations which could not solve any set:

Techniques	Iterations	Constraints	Time (sec.)	Solved
CD, UB, HTS and ITR B	64.9	9515.2	1038.9	128
CD, HTS and KI	75.3	9713.7	1023.5	128
CD, UB and ITR B	64.9	9442.5	1011.7	128
CD, UB and HTS	73.3	9383.4	953.5	128
CD, UB, HTS and KI	72.9	9406.1	989.8	128
CD, UB, HTS, KI and ITR A	57.5	8294.7	875.1	128
CD, UB, KI and ITR B	48.3	10072.6	833.8	256

Table 6: Results Benders Decomposition for the SMLRP

Cut disaggregation combined with the upper bound heuristic, knapsack inequalities and the incumbent B trust region accelerating techniques works the best. Furthermore, it is obvious that the IMIP method works much better than Benders decomposition for medium sized instances. Also for larger instances, Benders decomposition is outperformed by the SMLRP-IMIP. The larger instances with 100 facilities and 16 scenarios cannot be solved within 48 hours with Benders decomposition with CD, UB (8 scenarios), KI and the ITR B accelerating techniques. The slave problem is solved in milliseconds, consequently the solution time is almost fully determined by the master problem. The problem seems to lay in the convergence of the algorithm. To check that the solution time cannot be drastically improved by increasing δ slightly, we increased δ to 10 for the CD, UB, KI and ITS B set. These instances are still solved with absolute gaps lower than 0.0001 and consequently the same computational solution times.

For the RMLRP, we repeated the SMLRP experiments, but with the cut disaggregation acceleration technique excluded, as that cannot be applied to the RMLPR. For the RMLRP none of the combinations of the accelerating techniques can solve the 128 scenario and 25 facilities instance set within the hour. The combination of UB, HTR and ITR A and the same combination but with ITR B could solve 10 out of the 15 instances within the hour. This was followed by the combination of UB and ITR B with 8 instances. All other combinations can solve 3 instances or less. We conclude that accelerated Benders decomposition does not work well for the RMLRP.

7.5 Comparison and Discussion of Algorithms

Our IMIP formulation works very well and can solve instances with 50 candidate facilities and more than 100 scenarios, easily within the hour. The SLMRP seems a bit easier to solve than the RMLRP.

Although we use the most common accelerating strategies and add some of our own, Benders decomposition is computationally slower than solving the IMIP with CPLEX. An advantage of the Benders decomposition compared to the IMIP is that generally less memory is required, as constraints are added iteratively. We expect that the success of our IMIP formulation compared to the Benders decomposition can be explained by the following reasons:

- Our mixed integer formulation is very efficient and has $\Theta(\sum_{d \in D} \left(|L^d|^2|N_C| + |L^d||A_I^d|\right))$ variables less than the more standard mixed integer formulation (see Theorem 1 and 2).
- 64 bits computer architectures are more and more common in recent years, increasing the maximum possible random-access memory that can be used from 2^{32} different values (3-4 GB) to 2^{64} values (18 EB, one EB = 10^9 GB). Much research in the past has been done with 32 bit computer architectures (or even older computers), which cannot address sufficient random-access memory to solve the IMIP for instances of practical size.
- Since the introduction of MIP solvers, there have been many improvements. According to Bixby (2015), MIP solvers have improved with a factor of 800.000 since 1991.

As Bixby (2002) notes, "Three orders of magnitude in machine speed and three orders of magnitude in algorithmic speed add up to six orders of magnitude in solving power. A model that might have taken a year to solve 10 years ago can now solve in less than 30 seconds." Because of these kind of developments, it may be possible that a MIP formulation is computationally faster for some problems for which Benders decomposition worked better in the past.

8 Case Study Dutch Railways

In this section, we use our algorithm on instances based on the Dutch Railways. We use a green field approach, where existing facilities of the Dutch Railways are kept out of scope. The goal of this section is not to advise the Dutch Railways or to compare our solution to the current facilities, but to show that we can solve instances of practical size with our model.

We first describe, how these case study instances are generated, followed by our experiments. We investigate the benefit of the interchanges, the stability of the solution as the number of scenarios increases, the expected value of perfect information, and the value of the stochastic or robust solution. Both measures are common measures within the stochastic programming literature (Birge and Louveaux, 1997). The expected value of perfect information is the value of the S/RMLRP minus the wait and see solution. It is a measure that assesses how valuable perfect

information about the future is. The value of the stochastic and robust solution, is the improvement of adding future scenarios compared to only using the current situation. The expected value for the current case can be evaluated by solving the second stage problem for all scenarios while using the opened facilities of the current situation. The value of the robust/stochastic solution is now the expected value of the current case minus the S/RMLRP optimal objective value. Furthermore, we compare the solutions of the SMLRP and RMLRP with each other.

8.1 Dutch Railways Based Instances

The Dutch Railway instances each have 59 end stations and 55 candidate facilities. Four end stations (Utrecht, Breukelen, Amsterdam and Schiphol) have been excluded, because a maintenance facility cannot be built at these locations. The facility cost are an estimation of the average yearly cost of land, the necessary infrastructure and the maintenance facility itself including all side buildings. Furthermore, we altered the facility cost based on which province the end station is located in. The cost are either decreased or increased dependent on the province average land price. We assume unlimited capacity for the facilities and unlimited interchange capacities for all end stations except Utrecht, Amsterdam and Schiphol, where interchanges are not possible. The interchange budget has a different value for each experiment, and will be described in detail for each experiment.

We used four basic line plans: the current situation (2015), an estimation of 2018, and two possibilities for approximately 2025. These basic line planning contain all the lines (97,97,99 and 100 lines) the rolling stock serving the line and an estimate of the number of yearly maintenance visits per line. The future scenarios, are based on the plan "Beter en Meer" (Prorail and NS, 2014), where the Dutch Railways intends to increase the frequency of the lines in the *Randstad* or *larger Randstad*. The Randstad is a megalopolis in the Netherlands consisting of the four largest cities and their surrounding areas. The larger Randstad also includes the cities Amersfoort, 's-Hertogenbosch and Eindhoven.

The rolling stock consist of all current rolling stock and the future rolling stock types (FLIRT, SNG and ICNG). For each rolling stock, a rough estimation is made for the deadheading cost per kilometer. This estimation is based on many components such as driver cost, energy consumption, an estimation on the average dead heading velocity and availability costs. The availability costs are based on the life cycle cost of the rolling stock and the cost passenger dissatisfaction due to the unavailability of the train. The cost of the interchanges are set at 10 euro per interchange.

Scenarios are made by picking a random basic line plan, and altering the number of maintenance visits and rolling stock type. For each line of the line planning a random number of maintenance visits is chosen based on the triangular distribution. The number of maintenance visits of the basic planning is the mode of this distribution. We assume that the number of maintenance visits can decrease with 32.5% and increase with 75%, due to uncertainty in the yearly number of maintenance visits of the train unit and the number of passengers using a certain line. A maximum of 20% of the rolling stock types of the lines can be swapped with each

other. To do that, we generate for both regional and intercity trains a separated integer list, with its rolling stock number in the list for every maintenance visit it makes. These lists and the lines are shuffled. Then we go through all train lines and with a 10% probability, we change the rolling stock type to the first different rolling stock type of the list, all previous items of the list are removed. The maintenance frequency of the line is divided by the maintenance frequency of the current rolling stock type and multiplied by those of the new rolling stock type. We stop when there are no more lines or the limit of 20% has been reached.

For the SMLRP, we also have to generate the probabilities p_d. These probabilities are generated uniformly at random between 0 and 1, followed by scaling the probabilities such that they sum up to 1.

8.2 Experiments

In this subsection, we do three kinds of experiments. We test the influence of the interchange budget, the influence of the number of scenarios and we determine the expected value of perfect information and the stochastic/robust solution. Before we can test the the influence of the interchange budget, we need to know how many scenarios those instances should have. Consequently, we start with some initial computational experiments.

We do those experiments with sets of 10 instances, increasing the number of scenarios by a factor 2 for each set. We fixed the interchange budget for every scenario at $0.75M$, where $M = \sum_{l \in L^{2015}} m_l^{2015}$, which is the number of maintenance visits in the current (2015) scenario. For the RMLRP instances, the most often opened facilities are Eindhoven and Almere Oostvaarders (below 32 scenarios), Eindhoven and Hilversum (64 up to 4096 scenarios) and 's-Hertogenbosch and Hoofddorp for 8192 or more scenarios. However, for each number of scenarios, there are always other solutions. We evaluated the mentioned three solutions, with 10 instances that have 65536 scenarios, the difference between the optimal solution values is at most 2%. This small difference explains why we get different solutions, even if we include many scenarios. We show these solutions in Figure 4, within the first black circle are the stations in the north Randstad (Hilversum, Hoofddorp, Almere Oostvaarders) and the stations in the south of the larger Randstad (Eindhoven, 's-Hertogenbosch) are in the second black circle. For the interchange budget experiment (Section 8.2.1) we use instances with 8192 scenarios which can be solved in approximately ten minutes.

For the SMLRP instances, the average cost is always the same independent of the number of scenarios, and Eindhoven and Almere Oostvaarders is opened the most often. From 8 scenarios onwards, Eindhoven and Almere Oostvaarders are opened for all 10 instances. However, increasing the scenarios still decreases the standard deviation of the optimal objective values. Because of time and stability reasons we use instances with 16 scenarios for the interchange budget experiments (Section 8.2.1), these instances can be solved in approximately 20 minutes.

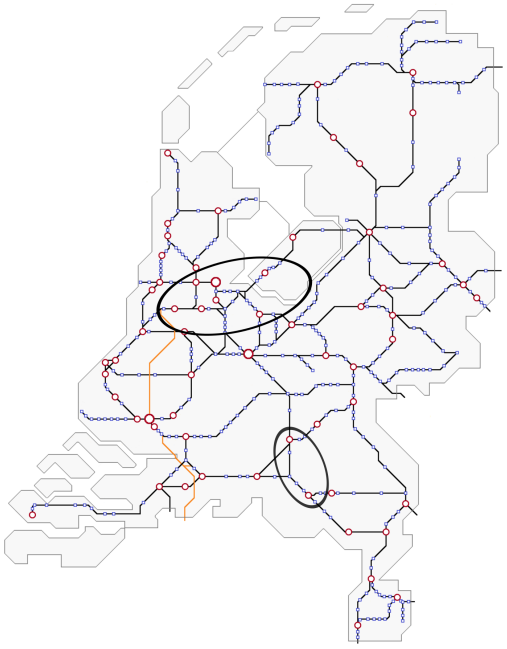


Figure 4: Railway map with all Dutch stations. The larger black circles are the best locations to open a maintenance facility. The small circles are the larger intercity stations and the tiny circles are the smaller regional stations. This picture is adapted from `https://upload.wikimedia.org/wikipedia/commons/6/6a/Spoorkaart_Nederland%2C_IC_stations.png`.

8.2.1 Influence of the Interchange Budget

We make 10 instances with 8192 (RMLRP) or 16 (SMLRP) scenarios according to the plan described in Section 8.1. For each of these instances, the budget is $G^d = 0, 0.25M, 0.5M, 0.75M, M, 2M$ and ∞ for all scenarios. Furthermore, we include 10 instances where the station interchange restrictions g_s are removed, while having an infinite interchange budget. These instances demonstrate the maximum gain that can be achieved by improving the stations interchange capacities.

Table 7 shows the influence of the interchange budget for the RMLRP. #sols, shows the number of different solutions while the opened facilities show the facilities which are opened the most often out of the 10 instances, followed by the number of times that they are opened. The average costs (optimal objective value) are shown in million euro's per year.

Two of the three earlier mentioned solutions come back in Table 7. 's-Hertogenbosch and Hoofddorp may be preferable for instances with a low budget, while Eindhoven and Hilversum may be preferable for instances with a high budget. In almost all cases, two facilities are opened, one in the north of the Randstad and one at the south of the large Randstad. The cost decrease with 15.7%, which is already reached with a budget of only $0.75M$. When we remove the station interchange restrictions, we can decrease the cost with another 10.2%.

Budget	#sols	Opened facilities	Cost (millions)
0	2	's-Hertogenbosch and Hoofddorp (8/10)	7.0
$0.25M$	4	's-Hertogenbosch and Hoofddorp (6/10)	6.3
$0.50M$	4	Eindhoven and Hilversum (5/10)	6.0
$0.75M$	3	's-Hertogenbosch and Hoofddorp (6/10)	5.9
M	7	Eindhoven and Hilversum (4/10)	5.9
$2M$	6	Eindhoven and Hilversum (4/10)	5.9
∞	6	Eindhoven and Hilversum (4/10)	5.9
No g_s	4	's-Hertogenbosch and The Hague (8/10)	5.3

Table 7: Budget interchange results for the RMLRP.

Table 8 shows the influence of the interchange budget for the SMLRP. Again all solutions consist of one maintenance facility in the south of the large Randstad and one in the north. From an interchange budget of $0.50M$ the solution is almost always Eindhoven and Almere Oostvaarders. The cost decreases with 25.4 %, already reached at M. Removing the interchange station capacity decreases the cost with another 13.6%. Furthermore, it can be seen that the SMLRP solution value is lower than the solution value of the RMLRP solutions.

Budget	#sols	Opened facilities	Cost (millions)
0	2	's-Hertogenbosch and Hoofddorp (6/10)	5.9
$0.25M$	3	Eindhoven and Hilversum (5/10)	5.2
$0.50M$	2	Eindhoven and Almere Oostvaarders (9/10)	4.7
$0.75M$	1	Eindhoven and Almere Oostvaarders (10/10)	4.5
M	1	Eindhoven and Almere Oostvaarders (10/10)	4.4
$2M$	1	Eindhoven and Almere Oostvaarders (10/10)	4.4
∞	1	Eindhoven and Almere Oostvaarders (10/10)	4.4
No g_s	3	Eindhoven and Almere Oostvaarders (8/10)	3.8

Table 8: Budget interchange results for the SMLRP.

8.2.2 Influence of the Number of Scenarios

We make sets of 10 instances with an interchange budget uniformly distributed between $0.25M$ and M, where we double the scenario for every set. Table 9 shows the influence of the number of scenarios for the RMLRP. Because the number of scenarios in Table 9 are very large, we only show the results in steps of factor 4 between 4 and 16384 scenarios; in all those cases the most common solution is Eindhoven and Hilversum.

Scenarios	#sols	Opened facilities	Cost (millions)	time
1	4	Eindhoven and Almere Oostvaarders (7/10)	4.5	0.1
2	5	Eindhoven and Almere Oostvaarders (5/10)	4.8	0.7
4	5	Eindhoven and Hilversum (5/10)	5.2	1.7
16	4	Eindhoven and Hilversum (7/10)	5.3	1.9
64	4	Eindhoven and Hilversum (7/10)	5.5	4.5
256	2	Eindhoven and Hilversum (8/10)	5.7	4.4
1024	4	Multiple solutions*	5.9	5.9
4096	3	Eindhoven and Hilversum (5/10)	6.0	6.5
16384	4	Eindhoven and Hilversum (5/10)	6.2	12.0
32768	3	's-Hertogenbosch and Hoofddorp (5/10)	6.3	9.7
65536	3	's-Hertogenbosch and Hoofddorp (7/10)	6.3	18.3

Table 9: Number of solutions, most common solution, cost in millions, and average time in minutes for different number of scenarios for the RMLRP.
*Both Eindhoven and Hilversum, and Hoofddorp and s-Hertogenbosch are chosen 4 times out of 10 as the solution.

For the RMLRP the solution remains somewhat chaotic, even for a high number of scenarios. As indicated in Section 8.2, this is probably because the solutions north of the Randstad in combination with Eindhoven and 's-Hertogenbosch are close to each other costwise. The cost increases with the number of scenarios, this is a property that generally holds for robust problems, as we always evaluate the cost of the worst scenario. The more scenarios we add, the higher the probability that an expensive scenario is added. The most common best robust solution is Eindhoven and Hilversum, but from 32768 scenarios onwards 's-Hertogenbosch and Hoofddorp is the most common solution. Furthermore, we can solve the RMLRP with SA extremely fast with only a 18.3 minute average solution time for instances with more than 65k scenarios.

Table 10 shows the influence of the number of scenarios for the SMLRP. The SMLRP solution becomes stable after 16 scenarios. Eindhoven and Almere Oostvaarders seems to be the best average case solution.

8.2.3 Expected value of Perfect Information and Stochastic/Robust Solution

The optimal objective for the wait and see solution (8192 scenarios), a solution for which the best facilities are opened for each scenario separately, is 6.1 million for the robust (minimax) objective and 4.6 million for the stochastic (average case) objective. This gives an expected value of perfect information of 3.2% for the RMLRP and 2.1% for the SMLRP.

The current case situation with a budget of 0.75M, has an optimal objective of 3.3 million euro per year. We evaluate this solution by solving the MRP for an instance with 65536 scenarios. In this case, the objective becomes 7.7 million euro per year for the robust objective and 5.4 million euro per year for the average case situation. Consequently, the expected value of the

Scenarios	#sols	Opened facilities	Cost (millions)	time
1	4	Eindhoven and Hilversum (5/10)	4.8	0.2
2	3	Eindhoven and Almere Oostvaarders (7/10)	4.6	0.4
4	2	Eindhoven and Almere Oostvaarders (7/10)	4.6	1.4
8	2	Eindhoven and Almere Oostvaarders (8/10)	4.7	4.4
16	1	Eindhoven and Almere Oostvaarders (10/10)	4.7	22.1
32	1	Eindhoven and Almere Oostvaarders (10/10)	4.6	79.8
64	1	Eindhoven and Almere Oostvaarders (10/10)	4.7	309.4

Table 10: Number of solutions, most common solution, cost in millions, and average time in minutes for different number of scenarios for the SMLRP.

robust solution is 18.2 % and 9.3 % for the stochastic solution.

We expect that the expected value of perfect information is low, because the combination of opening a maintenance facility in the north and south Randstad (see Figure 4) is a good solution for all scenarios . When the solution is not optimal, it is generally about 2% from optimality. Consequently, we know that we have found a solution which is robust for all scenarios, and that investing in better predictions in this case can only lead to small improvements. The value of the robust and stochastic solution shows that although there is a set of solutions which is good for all scenarios, just solving a scenario and using that solution is a bad idea. Consequently, we need the scenarios to find and verify good solutions for the R/SMLRP.

9 Conclusion

We formulated two novel models for the maintenance location routing problem and used different algorithms to solve them. Our IMIP formulation and the scenario addition method, work quite well and outperform Benders decomposition computationally. The SA method performs computationally better than the IMIP for the RMLRP and requires less memory as it only has to consider a subset of the scenarios. An advantage of the Benders decomposition compared to the IMIP is that generally less memory is required, as constraints are added iteratively. We expect that the success of our IMIP formulation compared to the Benders decomposition can be explained by the fact that it is a very efficient formulation which requires $\Theta(\sum_{d\in D}\left(|L^d|^2|N_C| + |L^d||A_I^d|\right))$ variables less than a standard mixed integer formulation (see Theorem 1 and 2), by using a 64 bits computer architecture and by the MIP solver improvements in recent years.

The case study indicates that we can solve instances of practical size, and that including scenarios can save 18.2% for the RMLRP and 9.3% for the SLMPR relative to using only the current line planning scenario. Including future scenarios will become even more important when facility capacity is a binding constraint. The current model includes a maximum capacity

for locations, but this has been set at infinity for the case study as the maximum capacity of a location is hard to estimate. Furthermore, the cost of a facility is only dependent on its maximum capacity in the current model. An interesting future direction is to make the price of the facilities dependent on the provided capacity. In such a case, solutions with (approximately) the same facility costs, can consist of many small facilities or a few larger facilities. Furthermore, there should be enough capacity for all scenarios, and ignoring some of the scenarios can lead to infeasible solutions. As it is costly to have enough capacity for all possible scenarios, even unlikely ones, we can also allow some recoverability. This recoverability can consist of building additional facilities, selling facilities or upgrading them to a higher capacity.

References

E. Álvarez-Miranda, E. Fernández, and I. Ljubić. The recoverable robust facility location problem. *Transportation Research Part B: Methodological*, 79:93–120, 2015.

Y. An and B. Zeng. Exploring the modeling capacity of two-stage robust optimization: Variants of robust unit commitment model. *IEEE Transactions on Power Systems*, 30(1):109–122, 2015.

Y. An, B. Zeng, Y. Zhang, and L. Zhao. Reliable p-median facility location problem: two-stage robust models and algorithms. *Transportation Research Part B: Methodological*, 64:54–72, 2014.

L. Anderegg, S. Eidenbenz, M. Gantenbein, C. Stamm, D. S. Taylor, B. Weber, and P. Widmayer. Train routing algorithms: Concepts, design choices, and practical considerations. In *Proceedings of the Fifth Workshop on Algorithm Engineering and Experiments*, volume 111, pages 106–118. SIAM, 2003.

J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1):238–252, 1962.

D. Bertsimas and J. N. Tsitsiklis. *Introduction to linear optimization*, volume 6. Athena Scientific Belmont, MA, 1997.

D. Bertsimas, E. Litvinov, X. A. Sun, J. Zhao, and T. Zheng. Adaptive robust optimization for the security constrained unit commitment problem. *IEEE Transactions on Power Systems*, 28 (1):52–63, 2013.

J. R. Birge and F. V. Louveaux. A multicut algorithm for two-stage stochastic linear programs. *European Journal of Operational Research*, 34(3):384–392, 1988.

J. R. Birge and F. V. Louveaux. *Introduction to Stochastic Programming.* Springer, 1997.

R. E. Bixby. Solving real-world linear programs: A decade and more of progress. *Operations Research*, 50(1):3–15, 2002.

R. E. Bixby. Optimization and analytics. Lunteren Conference on the Mathematics of Operations Research., 2015. URL `http://www.lnmb.nl/conferences/2015/programlnmbconference/LNMB-NGB_Bixby.pdf`.

T. C. Chan, Z.-J. M. Shen, and A. Siddiq. Robust facility location under demand location uncertainty. *arXiv preprint arXiv:1507.04397*, 2015.

L. Clarke, E. Johnson, G. Nemhauser, and Z. Zhu. The aircraft rotation problem. *Annals of Operations Research*, 69:33–46, 1997.

A. M. Costa. A survey on benders decomposition applied to fixed-charge network design problems. *Computers & Operations Research*, 32(6):1429–1450, 2005.

M. S. Daskin. *Network and Discrete Location: Models, Algorithms, and Applications*. Wiley, New York, 1995.

Z. Drezner and H. W. Hamacher. *Facility location: applications and theory*. Springer Science & Business Media, 2001.

T. A. Feo and J. F. Bard. Flight scheduling and maintenance base planning. *Management Science*, 35(12):1415–1432, 1989.

V. Gabrel, M. Lacroix, C. Murat, and N. Remli. Robust location transportation problems under uncertain demands. *Discrete Applied Mathematics*, 164:100–111, 2014.

R. Gopalan. The aircraft maintenance base location problem. *European Journal of Operational Research*, 236(2):634–642, 2014.

R. Jiang, M. Zhang, G. Li, and Y. Guan. Benders decomposition for the two-stage security constrained robust unit commitment problem. *Online Optimization*, 2011.

M. Khatami, M. Mahootchi, and R. Z. Farahani. Benders decomposition for concurrent redesign of forward and closed-loop supply chain network with demand and return uncertainties. *Transportation Research Part E: Logistics and Transportation Review*, 79:1–21, 2015.

C. Lee, C. Liu, S. Mehrotra, and Z. Bie. Robust distribution network reconfiguration. *IEEE Transactions on Smart Grid*, 6(2):836–842, 2015.

K. T. Lieckens, P. J. Colen, and M. R. Lambrecht. Optimization of a stochastic remanufacturing network with an exchange option. *Decision Support Systems*, 54:1548–1557, 2013.

T. L. Magnanti and R. T. Wong. Accelerating benders decomposition: Algorithmic enhancement and model selection criteria. *Operations Research*, 29(3):464–484, 1981.

G. Maróti and L. Kroon. Maintenance routing for train units: The transition model. *Transportation Science*, 39(4):518–525, 2005.

G. Maróti and L. Kroon. Maintenance routing for train units: The interchange model. *Computers & Operations Research*, 34(4):1121–1140, 2007.

M. T. Melo, S. Nickel, and F. Saldanha-da Gama. Facility location and supply chain management–a review. *European Journal of Operational Research*, 196(2):401–412, 2009.

G. Nagy and S. Salhi. Location-routing: Issues, models and methods. *European Journal of Operational Research*, 177(2):649–672, 2007.

F. Piu and M. G. Speranza. The locomotive assignment problem: a survey on optimization models. *International Transactions in Operational Research*, 21(3):327–352, 2014.

Prorail and NS. Beter en meer, 2014. URL `https://www.rijksoverheid.nl/documenten/rapporten/2014/03/28/bijlage-1e-operationele-uitwerking-prorail-en-ns-beter-en-meer`.

J. A. Rappold and B. D. Van Roo. Designing multi-echelon service parts networks with finite repair capacity. *European Journal of Operational Research*, 199:781–792, 2009.

E. D. Santibanez-Gonzalez and A. Diabat. Solving a reverse supply chain design problem by improved benders decomposition schemes. *Computers & Industrial Engineering*, 66(4):889–898, 2013.

T. Santoso, S. Ahmed, M. Goetschalckx, and A. Shapiro. A stochastic programming approach for supply chain network design under uncertainty. *European Journal of Operational Research*, 167(1):96–115, 2005.

A. Sarac, R. Batta, and C. M. Rump. A branch-and-price approach for operational aircraft maintenance routing. *European Journal of Operational Research*, 175(3):1850–1869, 2006.

Z.-J. M. Shen, R. L. Zhan, and J. Zhang. The reliable facility location problem: Formulations, heuristics, and approximation algorithms. *INFORMS Journal on Computing*, 23(3):470–482, 2011.

L. V. Snyder. Facility location under uncertainty: a review. *IIE Transactions*, 38(7):547–564, 2006.

C. Swamy and D. B. Shmoys. Approximation algorithms for 2-stage stochastic optimization problems. *ACM SIGACT News*, 37(1):33–46, 2006.

K. T. Talluri. The four-day aircraft maintenance routing problem. *Transportation Science*, 32 (1):43–53, 1998.

H. Üster and H. Agrahari. A benders decomposition approach for a distribution network design problem with consolidation and capacity considerations. *Operations Research Letters*, 39(2): 138–143, 2011.

J. C. W. van Ommeren and A. F. Bumb. Locating repair shops in a stochastic environment. *Computers & Operations Research*, 33:1575–1594, 2006.

R. M. van Slyke and R. Wets. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17(4):638–663, 1969.

B. Zeng and L. Zhao. Solving two-stage robust optimization problems using a column-and-constraint generation method. *Operations Research Letters*, 41(5):457–461, 2013.

L. Zhao and B. Zeng. Robust unit commitment problem with demand response and wind energy. In *2012 IEEE power and energy society general meeting*, pages 1–8. IEEE, 2012.

A Proof Proposition 2

We show that the MRP has an integer solution $z_l(a)$ if g_s ($\forall s \in S$), G, m_l ($\forall l \in L$) and q_n ($\forall n \in N_F$) are integer. The proofs are by contradiction. We show that every feasible fractional solution is a convex combination of feasible integer solutions, while we know from LP theory (Bertsimas and Tsitsiklis, 1997), that a basic feasible solution can never be a convex combination of two other feasible solutions.

Lemma 1. *If g_s ($\forall s \in S$), G, m_l ($\forall l \in L$) and q_n ($\forall n \in N_O$) are integer, then $\sum_{l\in L} z_l(a)$ ($\forall a \in A_F$) will be integer for any basic feasible solution.*

Proof. Because the total flow departing from the source, $\sum_{l\in L} m_l$, is integer and because of flow conservation, $\sum_{l\in L} z_l(a)$ is integral or there are multiple fractional sum flows. In the case there are multiple fractional sum flows, constraints (1), (5) or (6) can only be binding when there are multiple fractional sum flows within the same set associated with the constraints: $\delta_{\text{in}}(n)$ ($\forall n \in N_O$), A_I or A_s ($\forall s \in S$). Without a binding constraint, we can always apply rounding to the fractional sum flows. When we have multiple fractional values, within the same set, we can round the values to integers without changing the number of flow within the set. This rounding is done by rounding the individual $z_l(a)$, in such a way that $\sum_{l\in L} z_l(a)$ $\forall a \in A_F$ stays the same and $\sum_{a\in A_F} z_l(a)$ $\forall l \in L$ as well. This is possible because of flow conservation and proves that any fractional sum flow is a convex combination of the rounding possibilities. □

Lemma 2. *If g_s ($\forall s \in S$), G, m_l ($\forall l \in L$) and q_n ($\forall n \in N_F$) are integer, then $z_l(a)$ ($\forall a \in A_F$) will be integer for any basic feasible solution where $\sum_{l\in L} z_l(a)$ is integer.*

Proof. When $\sum_{l\in L} z_l(a)$ is integer, then all $z_l(a)$ are integer or multiple $z_l(a)$ are fractional. When there are multiple fractional $z_l(a)$, we can round some up and some down to an integer value, in such a way that $\sum_{l\in L} z_l(a)$ does not change. Because m_l is an integer, and we have flow conservation, we know that there is another arc, where the same type of flow is fractional. At this node we can do the opposite rounding, guaranteeing that also the amount of flow for

every type ($\sum_{a \in A_F} z_l(a)\ \forall l \in L$) does not change. The rounding proves that every fractional solution is a convex combination of the rounding possibilities. □

Combining these lemma's give us Theorem 2.

Acknowledgements

This study was funded by Nedtrain. The authors would like to thank Nedtrain and NSR. Furthermore, we want to thank Mei Li Kho, for gathering the current case (2015) data for the case study, and Bob Huisman and Geert-Jan van Houtum for giving valuable input.