Optimization and simulation for robust railway rolling-stock planning

S. Trefond a, b, *, A. Billionnet c, S. Elloumi d, H. Djellab e, O. Guyon a

a Innovation & Research, SNCF, Paris, France
b CEDRIC-CNAM, Paris, France
c CEDRIC-ENSIIE, Paris, France
d CEDRIC-CNAM and UMA-ENSTA, Paris, France
e Axe TGV Atlantique, SNCF Mobilites, Paris, France

ABSTRACT

In this paper, we focus on the problem of robust rolling-stock planning for French passenger trains. First, we characterize robustness and define some indicators for the evaluation of rolling-stock rosters. We take a particular interest in homogenizing turning-times in a roster in order to absorb potential delays. Then, we propose a new approach to solve the problem of robust rolling-stock planning. The SNCF reference tool (PRESTO) calculates a solution to the rolling-stock planning problem. It consists of a multi-step approach to cover demand while minimizing operating costs, and to further add maintenance slots to the roster. We propose an integrated ILP model to add robustness to a roster while maintaining low operating costs compared to PRESTO. We have carried out tests on nine real French regional transport instances, and we use a simulation module to evaluate the results. We observe a significant improvement in robustness indicators while maintaining low operating costs and meeting maintenance requirements.

© 2017 Elsevier Ltd. All rights reserved.

1. Introduction

Huisman et al. (2005) describe the three major tasks of railway operations management:

- Timetabling;
- Rolling-stock circulation planning to cover timetable;
- Crew scheduling to operate the rolling-stock.

These tasks are interdependent, but carried out separately by railway operators, usually in cooperation with the infrastructure manager. This paper deals with robustness issue in rolling-stock planning, which takes place from one year to six months before operations. We point out that building a robust transportation plan requires a collective work of all the actors. Punctuality and reliability are essential elements of passenger transportation systems. But disturbances such as infrastructure failures and rolling-stock breakdowns can occur, and then require an adaptation of the timetable, rolling-stock and

* Corresponding author. SNCF Innovation & Recherche, 40, avenue des Terroirs de France, 75611 Paris Cedex 12, France.
E-mail address: sabine.trefond@sncf.fr (S. Trefond).

http://dx.doi.org/10.1016/j.jrtpm.2017.02.001
2210-9706/© 2017 Elsevier Ltd. All rights reserved.
crew schedules. The aim of this study is to anticipate those potential disturbances by proposing robust rolling-stock rosters. A robust rolling-stock roster should anticipate operational disturbance possibilities in order to limit service quality deterioration and additional costs.

In this work, we take a special interest in the robust rolling-stock planning problem applied to passenger trains in France. The rolling-stock planning problem consists in providing the minimal number of rolling-stock units to cover a set of trains to operate. Several technical constraints have to be respected, such as maintenance requirements. For instance, each unit has to go to a maintenance depot every three days. However, in case of disturbances, optimized rolling-stock rosters usually make delays spread into the traffic network. Then, robust approaches enable roster to absorb delays. We propose an integrated method to build robust rolling-stock rosters while taking into account technical constraints such as maintenance requirements.

We first define robustness. We introduce indicators inspired by planning rules in railways to characterize robustness. In particular, we provide an indicator based on the following statement: homogeneous turning times bring robustness to a rolling-stock plan. Then, we propose a method to take into account this robustness indicator while building rolling-stock rosters. It allows us to compute robust solutions by working on their structure. Furthermore, our approach integrates maintenance requirements. Finally, we present optimization and simulation results, which show the relevance of the indicator and method.

Section 2 reviews robust approaches found in the literature. Then, Section 3 is dedicated to the description of the rolling-stock planning problem, while Section 4 describes an existing solution at SNCF. We focus on robustness indicators in railways. In Section 5, we present a definition of robustness in the studied context. We detail the indicators chosen to build and to evaluate robustness of rolling-stock rosters. Section 6 describes an ILP model of the robust rolling-stock planning problem that we address. Section 7 presents optimization tests and results on real instances. Finally, Section 8 describes a simulation process to show the relevance of the method.

2. Literature review

Railway operations management includes timetabling, rolling-stock planning, and crew scheduling (Huisman et al., 2005). Rolling-stock planning is a crucial part of this transportation planning process and has been frequently studied (Schrijver, 1993). Railway operations management also implies many underlying maintenance problems, such as the rolling-stock maintenance planning problem and the rail inspection scheduling problem (Peng et al., 2013). Maintenance on rolling-stock is often considered as technical constraints to respect in rolling-stock planning models (Cacchiani et al., 2010; Giacco et al., 2014). In this paper, we focus on the rolling-stock planning problem, and we integrate rolling-stock maintenance requirements.

Robustness has to be considered at every step of the planning process. All the resources of the transportation system have an effect on the system's robustness: times, rolling-stock, crew, stations, even signaling system (Hofman and Madsen, 2005; Landex and Jensen, 2013; Goverde et al., 2013). It is also considered during real-time operations, usually as rescheduling problems in case of disturbances (Larsen et al., 2014). Many criteria come under consideration regarding railway operations, such as costs, performance, quality of service, and robustness. These criteria sometimes are antagonistic, then transportation planning is a matter of tradeoff. For instance, Abril et al. (2008) study the tradeoff between capacity (performance) and robustness, Vansteenwegen et al. (2016) look at the tradeoff between quality of service and robustness, and Liden and Joborn (2016) evaluate the tradeoff between performance and cost.

In this paper, we tackle robustness related to rolling-stock planning in conception, and we take a special interest on the tradeoffs between costs, maintenance, and robustness.

Deterministic optimization approaches do not take into account uncertainty of the data. Under this hypothesis, an optimal solution applied in real conditions might be deteriorated in case of unexpected circumstances, or even become infeasible. Ben-Tal and Nemirovski (2000) came up to this conclusion: “In real-world applications of Linear Programming, one cannot ignore the possibility that a small uncertainty in the data can make the usual optimal solution completely meaningless from a practical viewpoint”.

Robust approaches exist to solve optimization problems while taking data uncertainty into account by modeling uncertainty as scenarios or intervals (Kouvelis and Yu, 1997; Soyster, 1973; Bertsimas and Sim, 2004). But classical robust approaches are not always adapted to industrial applications (Le Maitre, 2008).

Smith and Johnson (2006) have studied the robust airline fleet assignment problem. They apply rules related to the context to act directly on the structure of the solution. The rule used is the station purity: the number of fleet types serving a given station should not exceed a specified limit. By using this rule to solve the fleet assignment problem, they obtain robust solutions from an operational viewpoint.

In railways, several robustness indicators have been found to evaluate robustness of a transportation plan, or to build robust solutions.

To evaluate an existing transportation plan, Nielsen et al. (2007) define indicators associated to the delay, e.g., average delay, or cumulative delay. Hofman and Madsen (2005) use two complementary indicators: the regularity is the percentage of on time trains, while the reliability measures the number of actual departures compared to the number of departures planned. At SNCF, Chandesris (2005) computes the back to normal time, which is the time interval between the appearance of
a disturbance and the departure of the first on time train after it. Takeuchi et al. (2007) take an interest in the passengers’ view point: they define a robustness indicator measuring the deterioration of the disutility of passengers when a disturbance occurs. This disutility is based on criteria such as waiting time, additional transfers, congestion rate. In this paper, we refer to these indicators as evaluation indicators.

To build robust timetables, Hofman and Madsen (2005) add buffer times to absorb small delays. While 2-min-margins make the regularity equal to 70%, 5-min-margins make it reach 90%. Vromans et al. (2006) homogenize headways by calculating indicators called Sum of Shortest Headway Reciprocals (SSHHR) and Sum of Arrival Headway Reciprocals (SAHR). A headway is the time interval at a given station or intermediate point between two successive trains on the same itinerary. Nielsen et al. (2007) provide robust solutions to the rolling-stock planning problem by working on turning times and composition changes. A turning time corresponds to the interval between the arrival of a rolling-stock unit in a station and its next departure. The longer turning times are, the more small delays can be absorbed. By increasing turning times, the authors reduce the propagated delay, but they may need more units. It is the price of robustness. Nielsen et al. (2007) also minimize the composition changes to limit the risk of delay. Indeed, forming a multiple unit requires coupling operations to couple units together, which can cause delays. Furthermore, they propose indicators to build robust solutions to the crew scheduling problem. For instance, limiting crew transfers reduces the cumulative delay up to 30%. In this paper, these indicators are called construction indicators.

A structural method allows to reproduce experts practical rules adapted to the context. It seems to be the most relevant approach to solve the rolling-stock planning problem in our study.

In literature, simulation methods are often used to evaluate robustness of railway solutions (Corman et al., 2014; Pouryousef and Lautala, 2015). Abril et al. (2008) define simulation as “the imitation of an operation of a real-world process or system over time. It is the representation of dynamic behavior of a system by moving it from state to state in accordance with well-defined rules. […] For train scheduling, simulation has often been used in combination with other methods, originating what could be defined as hybrid models”. In this work, we use a simulation process to evaluate robustness of rolling-stock planning solutions.

3. Rolling-stock planning problem definition

We define a task $T_i$ as a non-splittable trip to be realized. It is characterized by departure and arrival stations $S_{D_i}^1$ and $S_{A_i}^1$, and departure and arrival times $D_{D_i}^1$ and $D_{A_i}^1$. For each task $T_i$, we know the demand, in general the number of passengers to transport, and the capacity, corresponding to the maximal number of rolling-stock units, defined later in this section, that can be used to cover the task. A more formal description of data related to tasks is provided in Section 6.1.

Example

In Fig. 1, we present a 6-task-timetable on a one-week-time period. Demands are given in brackets, and capacities in square brackets.

To cover tasks, it is possible to provide one or more rolling-stock units. A rolling-stock unit is a set of rail coaches that cannot be divided. Two or more units can be coupled to create a multiple unit, so that it can cover a higher demand task. A unit can be assigned to two successive tasks $T_i$ and $T_j$ if $T_j$ starts from arrival station of $T_i$, and if the turning time between the two tasks is greater than a technical threshold that is specific to each station. A turning time is the time between the arrival time of a task and the departure time of the next task covered by the same unit. More precisely, the turning time between tasks $T_i$ and $T_j$ is equal to $D_{D_i}^1 - D_{D_j}^1$.

A unit is assigned to a sequence of tasks called row of a unit. A rolling-stock roster is a sequence of rows. A unit can accomplish two successive rows $k_i$ and $k_j$ during two successive periods $p$ and $p + 1$ if $k_j$ follows $k_i$, that is to say if the unit can cover successively the last task of $k_i$ in period $p$ and the first task of $k_j$ in period $p + 1$.

A roster of a number of NU units is cyclic if the rows can be numbered from 1 to $NU$, so that for all $i$ from 1 to $NU - 1$, row $k_{i+1}$ follows row $k_i$, and row $k_1$ follows row $k_{NU}$. Then, the first week, each unit $u_i$ can be assigned to row $k_i$. The week after, for all i from 1 to $NU - 1$, unit $u_i$ covers row $k_{i+1}$, and unit $u_{NU}$ covers row $k_1$.

In addition, rolling-stock units must respect maintenance requirements. Based on the traveled distance or on the number of operating days, each unit needs to make periodic visits to a specified maintenance site called depot.

For a set of tasks, a feasible solution to the rolling-stock planning problem consists of a cyclic rolling-stock roster in which each task is covered and technical operating constraints are respected. Usually, the objective of the rolling-stock planning problem is to minimize the operating costs defined hereafter.

Dead-headings can be added to the roster in order to move units from a station to another. A dead-heading is a trip with no passengers. These trips may be necessary to cover every task. Moreover, they are necessary to move units to or from the depot for maintenance requirements. However, dead-headings cause additional costs for the company and increase the rail traffic in the network. In particular, they are sometimes not possible because of track occupation by other train services. An alternative, but more economical, way to move units consists in using more units than needed to cover a task. For one task, the units that cover the demand are active, additional ones are passive.
To sum up, a unit is assigned to a row, which is a sequence of tasks and dead-headings. When the unit is assigned to a task, it is considered as an active or as a passive unit.

Costs related to a unit are the number of kilometers that it travels: active costs of a unit correspond to the number of kilometers traveled as an active unit, while passive costs correspond to the number of kilometers traveled as a passive unit. The total number of units used and the active costs are called primary costs. Costs related to dead-headings and passive costs are called secondary costs. Operating costs include primary and secondary costs. Both of them are to be minimized. However, the impact of secondary costs is much lower than the impact of primary costs.

Example continued

To cover demands of the previous timetable (cf. Fig. 1), we propose an example of 5-row-roster in Fig. 2. Each unit can transport 200 passengers, and the time period is one week. On Week 1, Unit 001 covers Row 1, which is composed of three tasks T1 from A to B, T2 from B to C, and T4 from C to B. Unit 004 also covers T1 to satisfy the demand of 400 passengers to transport. Task T4 is covered by three units, even though two units would have satisfied the demand: the extra unit is passive, it is coupled to other units to reach station B, where it is needed. For this first time period, Unit 001 covers a single row (Row 1), while Row 1 is only covered by one unit (Unit 1). Similarly, each row from Row 2 to Row 5 corresponds to a single unit: Unit 2 to Unit 5. It shows the bijection between units and rows for one time period.

4. PRESTO: the rolling-stock planning tool at SNCF

Marcos (2006) has developed a software (PRESTO) to solve the rolling-stock planning problem at SNCF. PRESTO is a multi-stage approach that minimizes the operating costs. It is illustrated on the left side of Fig. 3.

PRESTO Stage 1 method consists of a MILP model corresponding to a multi-commodity flow problem. It computes on each task the number of active and passive units (flows), and creates dead-headings, so that the number of tasks covered is maximal, while the operating costs are minimal. As mentioned in Section 3, the operating costs are composed of the number of units, the active costs, the passive costs, and the dead-headings. PRESTO minimizes the number of units as a primary objective, while other costs are secondary. Stage 2 builds a roster in three steps: building rows, linking rows together to build a cyclic roster, integrating maintenance operations to that roster. Then, we obtain a cyclic rolling-stock roster with maintenance slots. Marcos et al. (2005) precisely describe the model used.

Our work focuses on finding a robust solution to the rolling-stock planning problem. We decided to keep the number of units computed by PRESTO. Indeed, using an additional unit is too expensive and will not be accepted in practice. The goal of

Please cite this article in press as: Trifond, S., et al., Optimization and simulation for robust railway rolling-stock planning, Journal of Rail Transport Planning & Management (2017), http://dx.doi.org/10.1016/j.jrtpm.2017.02.001
our work is to build robust solutions to the rolling-stock planning problem by integrating robustness indicators while limiting the deterioration of the secondary operating costs.

Then, we decided to integrate a robust module in PRESTO (cf. right side of Fig. 3). Stage 1 remains unchanged, while Stage 2 is replaced by a robust one, in order to build robust rolling-stock rosters with maintenance. In Stage 2, maintenance tasks are integrated once the roster rows have been built. The robust module builds rows of the roster by optimizing robust indicators while adding maintenance tasks.

5. Robustness integration

In this paper, we focus on robustness in rolling-stock planning. As mentioned in Section 4, the output of PRESTO Stage 1 is considered as input data for our model.

As explained in Section 2, we do not consider robustness by modeling data uncertainty. We have analyzed the characteristics of robust solutions, to identify relevant indicators. Then, our approach consists of a structural method. Such a method affects the structure of a solution by optimizing indicators to make it robust.

Throughout this paper, we shall consider that a robust roster should resist, limit delay propagation, or be easily recoverable, when a “weak disturbance” occurs. This refers to the following definition:

- "Resistance": ability to absorb small delays immediately without any change, so that there is no impact on the transportation plan;
- "Limitation of delay propagation": ability to absorb small delays, to limit their propagation to the entire transportation system;
- "Recoverability": ability to be easily "repaired" by measures (or management scenarios) that can solve or limit the delay propagation when facing a specific disturbance.

In this paper, we focus on "resistance" and "limitation of delay propagation" of the transportation plan, which can be managed during tactical planning.

According to this definition, a transportation planning is robust if it limits delays in case of disturbances. To formalize robustness of rolling-stock rosters, we had to determine robustness indicators. We studied two categories of indicators: evaluation indicators, and construction indicators. Evaluation indicators measure the robustness of an existing transportation plan, they reflect the true goals that we would like to reach to make a planning robust. Construction indicators may be used as heuristic vehicles to reach that goals, they are optimized to build robust transportation plans.

We define evaluation indicators used to measure robustness of a roster.

First, we can quantify the workload homogeneity of a roster. The workload of a single rolling-stock unit during a time period (e.g. a week) is the sum of all tasks durations (in minutes). We use an indicator based on the homogeneity of units workloads. Indeed, the more a unit travels, the higher is the probability to have an incident occurring. The homogeneity of units workloads is represented by the standard deviation of the workloads in a rolling-stock roster (whom). This indicator is...
defined for a set number of rolling-stock units and tasks: to compare the workload homogeneity of two different solutions makes sense if the whole workload is fixed.

Then, we define two indicators based on delays:

- Regularity: percentage of on-time arrivals;
- Spread delay: sum of departure delays due to previous delays.

Train delays are known a posteriori, they cannot be measured before running a transportation plan. On the contrary, margins added to the arrival time of a train are known a priori, and usually improve robustness. We can use this indicator to build robust transportation plans.

Then, to integrate robustness in the rolling-stock planning process, we define a construction robustness indicator based on turning times: turning times homogenization ($thom$).

Turning times are essential to build robust plans. On one hand, the shorter they are, the more delays spread. If a unit stands longer in a station, delay may be absorbed easily, thus having less impact on the rest of the transportation plan. On the other hand, long turning times may fill station capacity. To absorb delays, we try to make turning times uniform. The corresponding indicator $thom$, to be minimized, is the sum of turning times reciprocals. Indeed, minimizing the sum of the reciprocals makes the corresponding values homogeneous (Vromans et al., 2006). This indicator makes sense for a constant sum of turning times in the roster: it would not be efficient to minimize $thom$ while maximizing the number of tasks covered, or while minimizing the number of units. Indeed, the model would tend to reduce the number of tasks covered or to increase the number of units, in order to improve $thom$.

To verify the relevance of this construction indicator, we measure robustness of rolling-stock rosters. We use a simulation process to propagate delays in a transportation plan, so that it is possible to calculate evaluation indicators.

We present here the indicators used in the model. However, more indicators have been studied. In particular, we have modeled an indicator regarding coupling and uncoupling operations, which makes the model more complex to solve, and deteriorates performances (Trefond, 2014).

6. Robust rolling-stock planning - the RRSP method

The problem that we consider consists in building a robust cyclic rolling-stock roster with maintenance slots for one time period. Robustness construction indicator has to be optimized. Then, the objective is a trade-off between costs and robustness: secondary costs (dead-headings, passive trips) may be deteriorated to build robust solutions, but the deterioration has to be controlled. In this section, we present the ILP model that we will use to solve the problem. As already mentioned, we take from PRESTO Stage 1 important parts of the solution built, especially the number of units and the active costs.

Please cite this article in press as: Trefond, S., et al., Optimization and simulation for robust railway rolling-stock planning, Journal of Rail Transport Planning & Management (2017), http://dx.doi.org/10.1016/j.jrtpm.2017.02.001
6.1. Problem data

We define here useful notations divided up into categories to describe the problem that we consider: problem constants, stations, tasks, dead-headings, and maintenance.

**General data:**
- $NU$: number of units (roster rows), indexed by $k$;
- $L$: length of the time period, e.g. one week in our problem; in this case, we consider the chainings between tasks on each day and also the chainings between successive days;
- $M$: minimal time between the end of a row and the beginning of the next one.

**Data related to stations:**
- $NS$: number of stations, numbered 1..$NS$, indexed by $s$;
- $tms$: minimal turning time at station $s$ (used to define virtual tasks in Section 6.2.1).

**Data related to tasks:**
- $NT$: number of tasks to cover, numbered 1..$NT$, indexed by $i, j$;
- $dem_i$: required number of units to cover task $i$ (deduced from the number of passengers);
- $cap_i$: maximal number of units on task $i$;
- $Sd_i$: departure station of task $i$;
- $Sa_i$: arrival station of task $i$;
- $Dd_i$: departure time of task $i$;
- $Da_i$: arrival time of task $i$.

**Data related to dead-headings:**
- $\mathcal{S}$: set of pairs of stations, between which there can exist a dead-heading;
- $Cws_s$: length of a dead-heading from station $s$ to station $s_0$, 0 if $s = s_0$ (kilometers);
- $Dws_s$: duration of a dead-heading from station $s$ to station $s_0$, 0 if $s = s_0$ (minutes).

**Data related to maintenance:**
- $depot$: maintenance site;
- $\mathcal{E}$: set of stations $s$, so that there exists a dead-heading from $s$ to the depot;
- $ds_{s,depot}$: duration of a dead-heading from $s$ to the depot, or from the depot to $s$ ($\forall s \in \mathcal{E}$) (minutes);
- $pm$: minimal time interval between two maintenance slots (minutes);
- $dm$: minimal duration of a maintenance slot (minutes);
- $ncmin$: minimal number of maintenance slots on a roster row;
- $ncmax$: maximal number of maintenance slots on a roster row.

6.2. Data preprocessing

From problem data described in section 6.1, we define new data to model the problem by an ILP.

6.2.1. Virtual tasks and the augmented set $\Gamma$

To build our model, we define virtual tasks. They do not correspond to a displacement or to real tasks to cover: they do not have a demand and a duration, and their capacity is infinite. We just use them in the model to identify the origin and the destination stations of each unit.

We have numbered the real tasks to cover from 1 to $NT$, and the stations from 1 to $NS$. We create $NS$ beginning virtual tasks numbered from $NT + 1$ to $NT + NS$ corresponding to each station at the beginning of the time period. Similarly, we create $NS$ ending virtual tasks numbered from $NT + NS + 1$ to $NT + 2NS$ corresponding to each station at the end of the time period. In this model, each unit starts at a station $s$ with a beginning virtual task $NT + s$, executes a sequence of real tasks, and arrives at a station $s'$ with an ending virtual task $NT + NS + s'$.

We define $\Gamma$ as the set of all pairs of real or virtual tasks $i, j$ that can be chained up directly by the same unit. The pair $(i, j) \in \Gamma$ if stations correspond and, for real tasks, if the turning time between $i$ and $j$ can be respected. More precisely:
• any pair of real tasks \(i, j\) can be chained up if \(S^j_i = S^j_i\) and \(D^j_i \geq D^j_i + tm_s\);  
• any real task \(j\) can follow a beginning virtual task \(NT + s\) if \(S^j_i = s\);  
• any ending virtual task \(NT + NS + s\) can follow a real task \(i\) if \(S^j_i = s\).

We also add the set of all pairs of real or virtual tasks \(i, j\) that can be chained up by the same unit using a dead-heading. More precisely:

• any pair of real tasks \((i, j)\) is in \(\Gamma\) if it is possible to insert a dead-heading from the arrival station of \(i\) to the departure station of \(j\). It means that \((S^j_i, S^j_i) \in \psi\) and \(D^j_i \geq D^j_i + tm_s + D_{ws_s}^j;\)
• for any beginning virtual task \(NT + s\), the pair \((NT + s, j)\) is in \(\Gamma\) if it is possible to insert a dead-heading from \(s\) to the departure station of \(j\). It means that \((s, S^j_i) \in \psi;\)
• for any ending virtual task \(NT + NS + s\), the pair \((i, NT + NS + s)\) is in \(\Gamma\) if it is possible to insert a dead-heading from the arrival station of \(i\) to \(s\). It means that \((S^j_i, s) \in \psi;\)

We need to compute the departure time of a unit. We denote by \(D^j_{NT+i,s}\) the departure time of a unit starting at station \(s\) and whose first real task is \(i\). A unit starts at station \(s\) through a beginning virtual task \(NT + s\). Then, it executes a real task \(i\), either directly from station \(s\), or from a different station \(s'\). In the latter case, a dead-heading is performed from \(s\) to \(s'\) with duration \(D_{ws_s}\). Let \(s'\) be the departure station of \(i\) \((s' = S^j_i)\). If \(s = s'\), then \(D^j_{NT+i,s} = D^j_i\) (the unit starts at the same time as task \(i\)). Otherwise, \(D^j_{NT+i,s} = D^j_i - D_{ws_s}\) (the unit starts at the same time as the dead-heading).

Similarly, we need to compute the arrival time of a unit. We denote by \(D^0_i,_{TN+i,NS+s}\) the arrival time of a unit whose last real task is \(i\) and arriving at station \(s\). A unit executes a last task \(i\) ending at station \(s\) \((s = S^j_i)\). Then, it ends at station \(s'\) through an ending virtual task \(NT + NS + s'\), either directly if \(s = s'\), or by performing a dead-heading from \(s\) to \(s'\) with duration \(D_{ws_s}\). Let \(s'\) be the arrival station of \(i\) \((s = S^j_i)\). If \(s = s'\), then \(D^0_i,_{TN+i,NS+s} = D^j_i\) (the unit ends at the same time as task \(i\)). Otherwise, \(D^0_i,_{TN+i,NS+s} = D^j_i + D_{ws_s}\) (the unit ends at the same time as the dead-heading).

### 6.2.2. Maintenance

To create maintenance slots, we consider that any unit can cover a maintenance slot after each real or beginning virtual task \(i\).

Let \(j\) be a real task. Let \(k\) be a unit chaining \(i\) and \(j\) \((i, j) \in \Gamma\). Then, \(k\) can cover a maintenance slot between \(i\) and \(j\) if:

• the time interval between the arrival time of task \(i\) and the departure time of task \(j\) is long enough to add a dead-heading from \(S^j_i\) to the depot, a maintenance slot of duration \(dm\), and a dead-heading from the depot to \(S^j_j;\)
• there is a time interval longer than \(\overline{pm}\) between previous maintenance slot covered by \(k\) and this maintenance slot.

Then, we define the two following sets: \(F_i\) and \(E_i\).

We define \(F_i\) as the set of tasks \(j\) so that a unit \(k\) can chain up \(i\) and \(j\), but no maintenance can be performed between \(i\) and \(j\):

\[
F_i = \left\{ j : (i, j) \in \Gamma, D^j_i < D^j_i + d_{s_i depot} + dm + d_{s_j depot} \right\}
\]

For instance, in Fig. 4, we assume that unit \(k\) covers task \(i\). Then, it may cover task \(j\) or task \(j'\). If \(k\) covers a maintenance slot after task \(i\), it cannot cover task \(j\), but it can still cover task \(j'\). So, task \(j\) belongs to \(F_i\), while task \(j'\) does not.

We define \(E_i\) as the set of tasks \(j\) that cannot be followed by a maintenance slot if \(i\) is followed by a maintenance slot. Indeed, a unit \(k\) should not cover any maintenance slot during a time \(\overline{pm}\) after covering a maintenance slot. In other words, \(E_i\) is the set of tasks \(j\) so that the beginning time of a maintenance slot after task \(j\) is greater than the ending time of a maintenance slot after task \(i\), and is lower than the ending time of a maintenance slot after task \(i\) plus the duration \(\overline{pm}\). More formally:

\[
E_i = \left\{ j = 1.NT, D^j_i + d_{s_i depot} + dm \leq D^j_i + d_{s_j depot} \leq D^j_i + d_{s_i depot} + dm + \overline{pm} \right\}
\]

In Fig. 5, we illustrate the definition of \(E_i\). Suppose a unit \(k\) covers a maintenance slot \(M1\) after a task \(i\). Then, it cannot cover other maintenance slots during a time \(\overline{pm}\): \(M2\) after task \(j\) and \(M3\) after task \(j'\) in Fig. 5, but it can still cover maintenance slot \(M4\) after task \(j'\). Then, \(j\) and \(j'\) belong to \(E_i\), while \(j''\) does not.
6.3. ILP model

We model the robust rolling-stock planning problem with maintenance (cf. Section 3) by an integer linear program (ILP).

6.3.1. Variables

We define the following decision variables:

\[ y_{k,ij} = \begin{cases} 1 & \text{if unit } k \text{ covers successively tasks } i \text{ and } j; \\ 0 & \text{otherwise.} \end{cases} \]

\[ x_{k,i} = \begin{cases} 1 & \text{if unit } k \text{ covers task } i; \\ 0 & \text{otherwise.} \end{cases} \]

\[ z_{k,i} = \begin{cases} 1 & \text{if unit } k \text{ covers a maintenance slot after covering task } i; \\ 0 & \text{otherwise.} \end{cases} \]

6.3.2. Objective function

Our ILP model is based on costs of an existing cost-optimal solution computed by PRESTO Stage 1 (cf. Fig. 3). It computes a new solution in order to improve robustness. Robustness is considered by optimizing the `thom` robustness indicator. However, the resulting criteria may be in conflict with operating costs minimization. In practice, it is unacceptable to degrade primary costs. Then, the objective function has to be a tradeoff between robustness and secondary costs. It is a weighted sum of operating costs, robustness indicator, and an operating criterion described below.

The objective function to be minimized is defined as follows:

\[
P_w \sum_{k=1}^{NU} \sum_{(i,j) \in \Gamma} C_{w_{S_i,S_j}} x_{k,ij} + P_{\text{thom}} \sum_{k=1}^{NU} \sum_{(i,j) \in \Gamma} \Delta_{k,ij} y_{k,ij} + P_{\text{tint}} \sum_{k=1}^{NU} \sum_{(D_r^t-D_t^r) \in \{\text{tint, tsup}\}} y_{k,ij}
\]

(1)

6.3.2.1. Secondary costs. The first term of (1) corresponds to the secondary operating costs. Secondary costs are composed of passive trips and dead-headings. Passive trips usually are negligible compared to dead-headings. Then, they do not appear in the model. In the objective function, costs related to a dead-heading linking two tasks \( i \) and \( j \) have a specific penalty, for instance its length \( C_{w_{S_i,S_j}} \), which is the number of kilometers of a dead-heading between station \( S_i^r \) and station \( S_j^d \).

6.3.2.2. Robustness indicator. The second term of the objective function is the value of the robustness indicator based on turning times. The turning time between two successive tasks \( i \) and \( j \) equals \( D_j^d - D_i^r \).

To integrate robustness in the solution, we want to homogenize turning times in the roster, so we minimize the indicator `thom` described in Section 5. By default, all turning times lower than 1 min will be considered as 1-min-turning times. Conversely, turning times higher than 60 min are not considered.

For a turning time between real tasks \( i \) and \( j \) chained up directly by a unit \( k \), we define:
For any pair of real tasks $i$ and $j$ linked by a dead-heading $W$, there are two turning times: one between $i$ and $W$, and one between $W$ and $j$. By default, $W$ is placed in the middle, so that both turning times are equal. So, we consider two equal turning times:

$$\Delta_{k,i,j} = \frac{1}{\max \left(1, D_{k,i}^j - D_{k,j}^i\right)} \quad \text{if } D_{k,i}^j - D_{k,j}^i \leq 60;$$

$$= 0 \quad \text{otherwise.}$$

6.3.2.3. Operating criterion. The third term of (1) corresponds to an operational criterion to consider. A unit should not stand in a station more than a threshold, generally 20 min for regional trains (TER), in order to avoid to fill the station capacity. However, if the turning time is greater than 40 min, it allows the unit to go to a depot and to come back. So we try to avoid intermediate turning times. We define an interval $[t_{inf}, t_{sup}]$, and any turning time in this interval is penalized.

6.3.2.4. Objective weight parameters. As described above, the objective function is a weighted sum of secondary costs, robustness indicator values, and an operating criterion. We define the following weights:

- $P_w$: weight of dead-headings in the objective function;
- $P_{thom}$: weight of the sum of turning times reciprocals in the objective function;
- $P_{tint}$: weight of the intermediate turning times in the objective function.

These parameters have to be set according to a tradeoff between robustness and costs. Dead-headings generate the most important costs, then the weight $P_w$ should be high enough to limit the increase of corresponding costs. If wanted, solutions can be computed with a non-compensatory weight, in order to ensure minimal costs. Then, the robustness weight should reflect the tradeoff between robustness and costs. The last parameter should be negligible compared to robustness and costs. To fix parameter values, we performed preliminary tests (cf. Section 7).

6.3.3. Constraints

6.3.3.1. Existence of a roster. The existence of a cyclic rolling-stock roster of $NU$ units without maintenance requires the respect of a few constraints:

$$\sum_{i=NT+1}^{NT+NS} x_{k,i} = 1 \quad k = 1..NU$$

$$\sum_{j:(j,j)\in \Gamma} y_{k,j,i} = \sum_{j:(i,j)\in \Gamma} y_{k,i,j} \quad k = 1..NU, i = 1..NT$$
The departure station of row 1 is the depot \( S_d \). In the latter case, its next task will be an ending virtual task. This is modeled by the following formulation: for any real task \( i \) and any unit \( k \), if there exists a task \( j_1 \) so that unit \( k \) chains up \( j_1 \) and \( i \), then there exists a task \( j_2 \) so that a unit \( k \) chains up \( i \) and \( j_2 \).

According to constraints (4), a real task \( i \) has to be covered by at least \( \text{dem}_i \) units. Constraints (5) assure that at most \( \text{cap}_i \) units cover \( i \).

Constraints (6) guarantee that the obtained roster is cyclic. For any \( k \) from 1 to \( NU \), the arrival station of row \( k \) is the same as the departure station of row \( k + 1 \) (or 1 if \( k = NU \)). Constraints (7) relate to time coherence between two successive rows: for any row \( k \) from 1 to \( NU \), the time interval between the arrival time of row \( k \) and the departure time of \( k + 1 \) (or 1 if \( k = NU \)) is greater than the minimal margin \( M \). This margin is fixed by an operational constraint.

Constraints (8) express variables \( x_{k,i} \) according to variables \( y_{k,j} \) for any real or beginning virtual task \( i \). Ending virtual tasks do not have successors. Then, constraints (9) define variables \( z_{k,i} \) for each ending virtual task \( i \).

Since our method is based on an existing solution, the station capacity constraint is already guaranteed. Then, we do not need to consider it in the ILP model.

### 6.3.3.2. Maintenance constraints

The operational maintenance requirements are expressed as follows: each unit \( k \) has to go to the depot \( n_{\text{cmi}} \) to \( n_{\text{cma}} \) times during the time period (usually 2 to 3 times a week), and stay there for \( dm \) minutes. In addition, unit \( k \) should not go to the depot during a time \( pm \) after its last maintenance slot.

We formulate it as constraints in the ILP model:

\[
\sum_{k=1}^{NU} x_{k,i} \geq \text{dem}_i \quad i = 1..NT \tag{4}
\]

\[
\sum_{k=1}^{NU} x_{k,i} \leq \text{cap}_i \quad i = 1..NT \tag{5}
\]

\[
x_{k,NT+NS+s} = x_{k \mod NU + 1,NT\mod s + s} \quad k = 1..NU, s = 1..NS \tag{6}
\]

\[
\sum_{i=N+NS}^{NT+NS} \sum_{j(i,j) \in \Gamma} D_{ij} y_{k \mod NU + 1,i,j + L} \quad \text{NT} + 1
\]

\[
- \sum_{j=NT+NS+1}^{NT+2NS} \sum_{i(i,j) \in \Gamma} D_{ij} y_{k,i,j} \geq M \quad k = 1..NU \tag{7}
\]

\[
x_{k,i} = \sum_{j(i,j) \in \Gamma} y_{k,i,j} \quad k = 1..NU, i = 1..NT + NS \tag{8}
\]

\[
x_{k,i} = \sum_{j(i,j) \in \Gamma} y_{k,i,j} \quad k = 1..NU, i = NT + NS..NT + 2NS \tag{9}
\]

Constraints (2) guarantee that any unit starts with a beginning virtual task. Constraints (3) assure spatio-temporal coherence. A unit assigned to a task \( i \) arrives at station \( S_i \). Then, whether it is assigned to a next task \( j \) which departure station \( S_j = S_i \), or it stays at station \( S_i \). This is modeled by the following formulation: for any real task \( i \) and any unit \( k \), if there exists a task \( j_1 \) so that unit \( k \) chains up \( j_1 \) and \( i \), then there exists a task \( j_2 \) so that a unit \( k \) chains up \( i \) and \( j_2 \).

According to constraints (4), a real task \( i \) has to be covered by at least \( \text{dem}_i \) units. Constraints (5) assure that at most \( \text{cap}_i \) units cover \( i \).

Constraints (6) guarantee that the obtained roster is cyclic. For any \( k \) from 1 to \( NU \), the arrival station of row \( k \) is the same as the departure station of row \( k + 1 \) (or 1 if \( k = NU \)). Constraints (7) relate to time coherence between two successive rows: for any row \( k \) from 1 to \( NU \), the time interval between the arrival time of row \( k \) and the departure time of \( k + 1 \) (or 1 if \( k = NU \)) is greater than the minimal margin \( M \). This margin is fixed by an operational constraint.

Constraints (8) express variables \( x_{k,i} \) according to variables \( y_{k,j} \) for any real or beginning virtual task \( i \). Ending virtual tasks do not have successors. Then, constraints (9) define variables \( z_{k,i} \) for each ending virtual task \( i \).

Since our method is based on an existing solution, the station capacity constraint is already guaranteed. Then, we do not need to consider it in the ILP model.
Constraints (10) express that a unit covering a maintenance slot after a task \( i \) also covers \( i \). Constraints (11) and (12) define lower and upper bounds of the number of maintenance slots on a row: each unit goes to the depot between \( nc_{\text{min}} \) and \( nc_{\text{max}} \) times. It guarantees that each unit has to go to the depot at the given frequency.

Constraints (13) express spatio-temporal coherence when a unit \( k \) covers a maintenance slot. Indeed, it assures that it will not cover any task starting before the return of \( k \) in a station. Constraints (14) forbid the covering of new maintenance slots in the next \( pm \) minutes (cf. sets \( F_t \) and \( E_t \) in Section 6.2.2), so that two successive maintenance slots are not too close.

Constraints (11) to (14) guarantee the required number of maintenance slots at regular intervals on one row. With this model, we can observe two successive rows such that the first row can end with maintenance and the second row can start with maintenance. In this case, a post-optimization algorithm could remove one of the successive slots. This has not been implemented for the experiments in Section 7 because it rarely occurs in the results.

To sum up, we model the robust rolling-stock planning problem by an ILP:

\[
\begin{align*}
P \quad \min & \quad (1) \\
\text{s.t.} & \quad (2) \text{ to } (14) \\
& y_{k,i,j} \in \{0,1\}, \quad \forall k=1..\text{NU}, \forall (i,j) \in \Gamma \\
& x_{k,i} \in \{0,1\}, \quad \forall k=1..\text{NU}, i=1..\text{NT} + 2\text{NS} \\
& z_{k,i} \in \{0,1\}, \quad \forall k=1..\text{NU}, i=1..\text{NT} + \text{NS}
\end{align*}
\]

Then, the RRSP method is based on the solution of the Integer Linear Program \([P]\).

### 7. Computational experiments

At SNCF, the time period is one week \((L=10080 \text{ minutes})\). We have run tests on nine real-world large instances from TER (French regional transport) involving 166 to 763 tasks and 5 to 15 rolling-stock units. These instances have been chosen to form a representative sample given the existing regions and characteristics such as the number of tasks, the number of rolling-stock units needed, etc. We compared the solutions of the new method (called RRSP) to PRESTO solutions. The model presented in Section 6.3 is solved with IBM ILOG CPLEX 12.3 on a 3.39 GHz Intel Core i7-2600 PC with memory of 8 GB and running Windows XP. CPLEX parameters are set to default, and the computed solutions are optimal (to the default MIP gap) for the nine instances.

#### 7.1. Instances

Each instance is characterized by:

- an ID number;
- the number of tasks to cover (maximal covering calculated by PRESTO);
- the number of units used by PRESTO (minimal number of units required for maximal covering).

#### 7.2. Parameters setting

We have run preliminary tests to observe twelve parameters values combinations on previous real-world instances. We fixed the values according to the results analyzed from the practitioners’ point of view:

- additional costs have to stay reasonable;
- robustness improvement has to be significant compared to the additional costs that it generates.

Then, parameters are set to the following values:

- dead-headings penalty \( P_w = 1500 \);
- turning times homogeneity penalty \( P_{\text{thom}} = 300 \);
- intermediate turning times penalty \( P_{\text{tint}} = 100 \);
- intermediate turning times interval \([t_{\text{inf}}, t_{\text{sup}}] = [20, 40] \);
- minimal time interval between two maintenance tasks \( pm = 2.25 \text{ days} \);
- minimal number of maintenance tasks on a row \( nc_{\text{min}} = 2 \);
- maximal number of maintenance tasks on a row \( nc_{\text{max}} = 3 \).
7.3. Results

Table 2 compares PRESTO solutions and RRSP solutions on test instances 1 to 9:

- **Instance**: reference to instances in Table 1;
- **Active costs**: active traveling distance (kilometers), calculated by PRESTO (the same for both methods);
- **Method**: method used to compute the solution (PRESTO or RRSP);
- **Time**: total computation time (minutes), we point out that RRSP method uses PRESTO Stage 1, which is included in the computation time;
- **Passive (operating costs)**: passive traveling distance (kilometers);
- **Dead-headings (operating costs)**: traveling distance corresponding to dead-headings (kilometers);
- **Total (operating costs)**: sum of active costs, passive costs, and dead-headings (kilometers);
- **thom** (robustness): value of the turning times homogeneity indicator (sum of turning times reciprocals), minimized in method RRSP;
- **whom** (robustness): standard deviation of all units workloads (minutes).

To make it easier to read, we highlighted key numbers. For each instance, we put in bold the best solutions in terms of robustness (thom and whom indicators). In average, RRSP degrades total operating costs of about 0.1%. It improves robustness indicators thom and whom of respectively 37% and 47%.

We should first note that the number of constraints (resp. binary variables) of the RRSP ILPs varies between 11,000 and 200,000 (resp. 74,000 and 3,000,000). Regarding Cplex solution time of RRSP, 10–50% of the time is dedicated to the solution of the linear relaxation at the root of the branch and bound.

Our approach allows robustness improvement considering the measured homogeneity indicators thom and whom.

We observed an average improvement of about 37% regarding the robustness indicator based on turning times (thom). As a result, the turning times are better-balanced in the whole roster and the workload is shared out smoothly between the units. Indeed, the average improvement of the corresponding indicator (whom) on the nine instances is about 47%. This phenomenon is illustrated for instance 1 in Fig. 6 by box plots, which show the dispersion of the units workload in PRESTO solution and RRSP solution. Each box plot has been built from the seven units workload values of a solution for instance 1. It presents three specific values (the maximum workload, the minimum workload, and the average workload), and an interval containing 50% of the values. Workloads in PRESTO solution vary between 1500 and more than 2600 min, and 50% of them are between 1600 and 2200, while workloads in RRSP solution vary between 1600 and 2400 min, and 50% of them are between 1800 and 2100 min. Then, it shows that workloads are more homogeneous in RRSP solution.

The number of units used and the active traveling distance are identical in PRESTO and RRSP solutions since they are determined in PRESTO Stage 1.

Dead-headings are penalized in RRSP solutions, so that we can observe an average enhancement of about 5% on the nine instances in Table 2 compared to PRESTO solutions. This can be explained by a deterioration of passive costs. Indeed, passive costs deterioration is not controlled in RRSP method. As a result, the costs can be significantly increased: for instance 7 in Table 2, there are 547 passive kilometers in the RRSP solution, compared to 343 km in PRESTO solution, corresponding to an increase of 60%. But the increase remains low compared to the global costs, since the deterioration of total operating costs is lower than 1%. For the nine instances, the total operating costs are still low, with an average gap of 0.1% between PRESTO and RRSP solutions. However, we present a decision aid module: the user may want to adjust the tradeoff between operating costs and robustness. Then, if dead-heading costs are too high for some instance (e.g. an increase of 3.6% for instance 2), they may be reduced by working on the weight $Pw$ of the objective function.

For any instance presented, solutions include maintenance whenever it is possible with the optimal number of units. For instances 1, 2, 4, 5, 6, and 8 in Table 1, PRESTO and RRSP solutions respect maintenance constraints. RRSP integrates maintenance slots while the rolling-stock roster is built-up, so that maintenance constraints are respected, and operating costs and

<table>
<thead>
<tr>
<th>Instance</th>
<th>Number of tasks</th>
<th>Minimal number of units computed by PRESTO (NU)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>207</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>314</td>
<td>11</td>
</tr>
<tr>
<td>3</td>
<td>597</td>
<td>15</td>
</tr>
<tr>
<td>4</td>
<td>166</td>
<td>11</td>
</tr>
<tr>
<td>5</td>
<td>241</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>352</td>
<td>7</td>
</tr>
<tr>
<td>7</td>
<td>372</td>
<td>6</td>
</tr>
<tr>
<td>8</td>
<td>482</td>
<td>15</td>
</tr>
<tr>
<td>9</td>
<td>763</td>
<td>15</td>
</tr>
</tbody>
</table>
robustness indicators are not deteriorated after the optimization phase. Regarding instances 3, 7, and 9, tests have been run without maintenance constraints with both methods. Indeed, PRESTO method is sequential, it integrates maintenance tasks once the roster rows have been built (cf. Section 4). Then, it is not always feasible to respect maintenance constraints. In our case, there is no feasible solution with maintenance for instances 3, 7, and 9.

8. Solution evaluation

To evaluate solutions, we develop a simulation module. It consists of a model to propagate delays, and we compute evaluation indicators for each solution.

8.1. Simulation module

First of all, we create a delay model (strata) based on the analysis of theoretical and realized timetables. We use a stratification method to divide trains into subgroups according to their characteristics:

- delay: no delay, a one-to 2-min delay, a two-to 5-min delay, etc.,
- time period, e.g. peak period,
- zone: the railway network is divided into geographical areas.

Table 3 shows an example of strata for one time-zone pair: 80% of the trains are on time, while 2% are 5–10 min late.
From these strata, we can generate scenarios by stratified sampling. A single scenario consists of a set of initial delays, where a delay is a number of minutes allocated to a train. We use a proportionate allocation strategy to generate representative scenarios: we allocate an initial delay to each task so that the delay distribution corresponds to the strata.

From a rolling-stock roster and strata, the simulation module propagates initial delays: for each train $T_i$ delayed, the delay spreads to the next train $T_j$ covered by the same unit, depending on the turning time between those two trains $D_{ij} - D_{ij}^d$. If the turning time is longer than the delay added to the minimal turning time $t_{min}$, the delay is completely absorbed. Else, the delay is partially absorbed, and the spread delay at the departure of $T_j$ is $D_{ij} - D_{ij}^d - t_{min}$. We can then compute the spread delay as the sum of departure delays due to previous delays. Fig. 7 illustrates the spread delay computation for a small example.

On Fig. 7, let a unit cover successively trains $T_1$, $T_2$, and $T_3$. The minimal turning time is 8 min. A 5-min initial delay is allocated to $T_1$. The theoretical turning time between $T_1$ and $T_2$ is 8 min, it cannot be reduced. Then, the initial delay is propagated to $T_2$. The theoretical turning time between $T_2$ and $T_3$ is 11 min, it can be reduced by 3 min. Then, the initial delay is partially absorbed. The total spread delay on this example is 7 min.

![Fig. 7. Initial delay propagation in a three-task example with an 8-min minimal turning time.](image)

<table>
<thead>
<tr>
<th>Table 3</th>
<th>Example of strata for one time-zone pair.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delay (minutes)</td>
<td>Frequency</td>
</tr>
<tr>
<td>0</td>
<td>80</td>
</tr>
<tr>
<td>0–2</td>
<td>11</td>
</tr>
<tr>
<td>2–5</td>
<td>6</td>
</tr>
<tr>
<td>5–10</td>
<td>2</td>
</tr>
<tr>
<td>&gt; 10</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 4</th>
<th>Simulation results to compare PRESTO and RRSP solutions on test instances 1 to 9.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance</td>
<td>Method</td>
</tr>
<tr>
<td>1</td>
<td>PRESTO</td>
</tr>
<tr>
<td></td>
<td>RRSP</td>
</tr>
<tr>
<td>2</td>
<td>PRESTO</td>
</tr>
<tr>
<td></td>
<td>RRSP</td>
</tr>
<tr>
<td>3</td>
<td>PRESTO</td>
</tr>
<tr>
<td></td>
<td>RRSP</td>
</tr>
<tr>
<td>4</td>
<td>PRESTO</td>
</tr>
<tr>
<td></td>
<td>RRSP</td>
</tr>
<tr>
<td>5</td>
<td>PRESTO</td>
</tr>
<tr>
<td></td>
<td>RRSP</td>
</tr>
<tr>
<td>6</td>
<td>PRESTO</td>
</tr>
<tr>
<td></td>
<td>RRSP</td>
</tr>
<tr>
<td>7</td>
<td>PRESTO</td>
</tr>
<tr>
<td></td>
<td>RRSP</td>
</tr>
<tr>
<td>8</td>
<td>PRESTO</td>
</tr>
<tr>
<td></td>
<td>RRSP</td>
</tr>
<tr>
<td>9</td>
<td>PRESTO</td>
</tr>
<tr>
<td></td>
<td>RRSP</td>
</tr>
</tbody>
</table>
8.2. Simulation results

Solutions have been optimized by a structural method (RRSP), taking into account a robustness indicator. We use the simulation module to compare the RRSP solutions to the PRESTO ones in terms of robustness. For each instance, 100 scenarios are generated. Indeed, preliminary tests showed that values of the computed indicators converge from 100 scenarios. For each scenario, the simulator computes the robustness indicators.

In Section 5, we define robustness as the ability to absorb small delays, so that there is no impact on the transportation plan. This means that delays should not spread too much. As a result, the cumulative spread delay should be lower in a robust roster. Regularity, which is the rate of on-time trains, is also an indicator of delay absorption. Indeed, the less delays spread, the more on-time trains there are.

Table 4 compares the average values of the indicators for each solution, and for each instance, we put in bold the best solutions in terms of robustness (Regularity and Spread delay):

- Instance: reference to instances in Table 1;
- Method: method used to compute the solution (PRESTO or RRSP);
- Regularity: percentage of on-time arrivals;
- Spread delay: sum of departure delays due to previous delays (format HH:MM);

On the nine reference instances, we observe an average improvement of 2% of regularity. Furthermore, the cumulative spread delay is reduced by almost 30 h, which represents a decrease of about 40% in average. This last indicator is only deteriorated for Instance 6, where the spread delay is 7 min larger in RRSP than in PRESTO. But it represents less than 2% of the global time of 6 h and 41 min. This can be explained by special properties. Indeed, some roster rows of Instance 6 can only be composed of tasks between two stations S and S’, because any task departing from (respectively arriving at) station S arrives at (respectively departs from) station S’. Since these rows have a high workload, the indicator thom has no impact on them. Then, the cumulative spread delay in Instance 6 depends on the initial delay distribution.

To conclude, we can say that the comparison of both methods on these indicators is significant: for each instance, the RRSP solutions are more robust than the PRESTO ones. Indeed, except for one instance, the spread delay is significantly decreased, while regularity is increased. These results validate the method that we propose to build robust rolling-stock rosters. Furthermore, it shows the relevance of the indicator based on turning times as a robustness construction indicator.

9. Conclusion

In this paper, we study a rolling-stock planning problem with a robustness perspective. First, we propose a definition of robustness and indicators to quantify robustness of rolling-stock rosters. We focus on a specific indicator to build robust solutions, which consists in homogenizing turning times. We present a method called RRSP and based on an ILP model to solve the rolling-stock planning problem. RRSP is an integrated approach: it improves robustness, keeps operating costs low, and takes into account maintenance requirements. Compared to usual sequential approaches, it prevents from sub-optimality of some criteria.

Then, tests have been run on nine representative real-world instances from the French regional passenger transport system. In the RRSP solutions, we observe an average improvement of about 37% and 47% regarding the robustness indicators respectively based on turning times and workloads, while total operating costs are barely deteriorated compared to the existing solution PRESTO at SNCF. Furthermore, we developed a simulation process to compute evaluation indicators based on delays. We observe an average improvement of 2% of regularity and 40% of spread delay in the RRSP solutions. These results validate the method used to build robust rolling-stock solutions, and show the relevance of the robustness indicator based on turning-times.

The RRSP module has been industrialized as part of a rolling-stock planning tool for French regional transport. The RRSP method can be applied to other types of trains (high speed trains, interregional trains, commuter trains, ...). Furthermore, some other indicators may be integrated to the model, such as the minimization of coupling and uncoupling operations.

References


Please cite this article in press as: Tréfond, S., et al., Optimization and simulation for robust railway rolling-stock planning, Journal of Rail Transport Planning & Management (2017), http://dx.doi.org/10.1016/j.jrtpm.2017.02.001