ROADEF/EURO 2014 Challenge
Trains don't vanish !

Rolling stock unit management on railway sites

François Ramond, Nicolas Marcos and Francis Sourd
SNCF - Innovation & Research Department

October 10, 2013

Contents

1 Introduction

The aim of this challenge is to find the best way to handle trains between their arrivals and departures in terminal stations. Today, this problem is shared between several departments at SNCF, so it is rather a collection of sub-problems which are solved in a sequential way. Consequently, the way we formulate it here, in an integrated way, is a prospective approach.

Between arrivals and departures in terminal train stations, trains never vanish. Unfortunately, this aspect is often neglected in railway optimization approaches. Whereas in the past, rail networks had enough capacity to handle all trains without much trouble, this is not true anymore. Indeed, traffic has increased a lot in recent years and some stations have real congestion issues. The current trend will make this even more difficult to deal with in the next few years.

In the proposed model, we focus on the multiple dimensions of the problem, in the sense that many different aspects are taken into account. We remain within geographically limited perimeters, typically a few km in urban areas: we consider a train station and the surrounding railway infrastructure resources. Solutions to such problems involve temporary parking and shuntings on infrastructure which are typically platforms in stations, maintenance facilities, rail yards located close to train stations and tracks linking them (these infrastructure resources constitute the "system").

One of the greatest challenges in the industry is to design software in a flexible and maintainable way, so that new constraints or objectives expressed by business users can be easily taken into account. To reflect this situation, some changes might be introduced in the initial subject at the end of the qualification phase. As a consequence, after analysis of the submitted results by experts, **the problem described in this document might change slightly for the final phase of the challenge**.

This document is organized as follows. A description of the problem is provided in Section 2. This desciption is necessary to understand the industrial problem; most concepts and notations are introduced in this section. Nevertheless, readers interested in a formal description of the problem might start directly with Sections 3 (Solution representation), 4 (Constraints) and 5 (Objectives), and refer to Section 2 or to the list of notations in appendix whenever needed. Sections in blue are not required to solve the problem, they only explain some modeling choices and give some insights in the industrial process.

2 Problem description

2.1 Planning horizon

The planning horizon considered in this problem is variable. It is an integral number ($nbDays$) of days from morning of day 1 ("d_1" at 00:00:00, denoted h_0) to midnight of day $nbDays$ ("d_{nbDays}" at 23:59:59). No absolute date is

considered, we assume that the days to be planned can be at any date. All days last 24 hours, and no time change occurs in the horizon. The set of all time instants within the planning horizon is denoted by $\mathcal{H}$. In the following, we use the word "time" to represent a time instant (date/time) during the horizon. A time within this horizon is written "$d_i\ hh : mm : ss$", where $i \in [\![1, nbDays]\!]$ stands for the day index, $hh \in [\![0, 23]\!]$ for the hours, $mm \in [\![0, 59]\!]$ for the minutes and $ss \in [\![0, 59]\!]$ for the seconds.

This representation implies that the time horizon is discretized, the smallest duration taken into account being one second. Durations have a format similar to time instants ($hh : mm : ss$), but the number of hours hh may be greater than 23.

Note that using the second as the shortest time unit enables to represent time with a high precision, considering that most durations used in the following are typically a few minutes or hours. All instances may not fully exploit this precision; some may only handle durations and time instants as multiples of 10 seconds or one minute, for example. Depending on the resolution approach, this might be used to reduce the problem complexity.

2.2 Arrivals

Arrivals are the end of journeys for passengers. In our model, arrivals generate entrances of trains in the system. The times of arrivals are provided as an input and are considered to be non-modifiable. These times correspond to the moments trains arrive on plaforms, but their entrance in the system usually occurs a few minutes before. Indeed, trains have to perform what we call "arrival sequences". These sequences are fixed and represent the routing of trains on the tracks during the last few km before platforms. Arrival sequences are non-modifiable but they have an impact on the system in the sense that they require some tracks during pre-defined time periods.

Formally, the set of arrivals during the horizon is denoted by $\mathcal{A}$ and an arrival $a \in \mathcal{A}$ is defined by the following characteristics:

- the associated train, $arrTrain_a$ (see the set $\mathcal{T}$ of trains introduced in Section 2.4),
- the arrival time $arrTime_a$,
- the arrival sequence $arrSeq_a$, which is the sequence of track groups (see Section 2.7.5) used immediately before arriving on platform,
- the set of preferred platforms, $prefPlat_a$, determining which platforms are preferred to be assigned to a (these preferences may be unsatisfied in solutions, but this is penalized by a cost in the objective function),
- the ideal dwell time, $idealDwell_a$, and the maximum dwell time, $maxDwell_a$, which respectively represent the ideal and the maximum time $arrTrain_a$ should stay on the platform after $arrTime_a$ before moving to some other resource,

- the remaining distance before maintenance (DBM), $remDBM_a$, and time before maintenance (TBM), $remTBM_a$, of $arrTrain_a$, determining whether or not $arrTrain_a$ must perform maintenance operations before being assigned to a departure (see description of maintenance in Section 2.6. In the following, if train $t \in \mathcal{T}$ is associated with a ($arrTrain_a = t$), we define $remDBM_t = remDBM_a$ and $remTBM_t = remTBM_a$.

Potentially, a can be part of a joint-arrival denoted by $jointArr_a$. A joint-arrival defines a combination of trains, physically assembled and arriving together at the station. More details on joint-arrivals and joint-departures are provided in Section 2.5.

Most decisions associated with arrivals are fixed and non-modifiable. The only decision to make concerning an arrival a is the platform to assign, i.e. the platform on which $arrTrain_a$ arrives at $arrTime_a$.

2.3 Departures

Departures are known by passengers as the beginning of a train journey. From the problem's perspective, departures are the way trains leave the system. As for arrivals, a platform must be assigned to each departure; their times as well as their departure sequences (routing between platforms and the limits of the system) are also fixed.

However, one has to decide which train is assigned to each departure. Assigning no train to a departure is highly undesirable; the number of unassigned departures constitutes the most significant criterion in the objective function (see section 5.1). Note that at most one train can be assigned to a departure. The train assigned to departure d, if any, is denoted by $depTrain_d$.

The following attributes define a departure $d \in \mathcal{D}$, where $\mathcal{D}$ represents the set of all departures:

- its departure time, $depTime_d$,
- its departure sequence, $depSeq_d$, which is the sequence of track groups (see Section 2.7.5) used immediately after departing from platform,
- the set of preferred platforms, $prefPlat_d$, determining which platforms are preferred to be assigned to d,
- the set of compatible train categories, $compCatDep_d$, which is a subset of $\mathcal{C}$ (see section 2.4.2) determining which trains can be assigned to d (only trains whose category is in $compCatDep_d$ can be assigned),
- the ideal dwell time, $idealDwell_d$, and the maximum dwell time, $maxDwell_d$, which respectively represent the ideal and the maximum time the assigned train should stay on the platform before $depTime_d$,
- the distance, $reqDBM_d$, and time, $reqTBM_d$, of the journey following the departure. These two values are compared, for a train $t \in \mathcal{T}$, with the

remaining DBM and TBM of t, to determine whether or not maintenance operations have to be performed on t before $depTime_d$.

Potentially, d can be part of a joint-departure represented by $jointDep_d$ (see section 2.5).

This description assumes that the assignments of trains to departures have no impact on arrivals and, more precisely, on the remaining TBM and DBM of trains associated with arrivals. In practice, this is not completely true because the trains associated with arrivals are usually trains which were earlier assigned to departures and which spent some time out of the system before coming back.

To take this into account, some arrivals are linked with departures occuring earlier in the horizon. For such an arrival $a \in \mathcal{A}$, we introduce an additional parameter denoted by $linkedDep_a \in \mathcal{D}$. $linkedDep_a$ is the departure whose assigned rolling stock unit comes back in the system associated with a.

Then, if a train t is assigned to d, the default train category of $arrTrain_a$ provided in the input data is replaced by the train category of t. In a similar way, the default remaining DBM and TBM of a are replaced by values depending on t (see Section 2.6).

2.4 Trains

We consider a set of trains denoted by $\mathcal{T}$. In our model, we define a train as a visit in the system of a rolling stock unit. The set of trains is composed of:

- trains already present in the system, located on one of its resources at the beginning of the horizon (this set is represented by $\mathcal{T}_I \subset \mathcal{T}$, see section 2.4.1), and
- trains associated with arrivals during the horizon (these trains belong to set $\mathcal{T}_A \subset \mathcal{T}$).

Rolling stock units are unoriented, composed of railcars which may not be decomposed nor recombined with those of other units. In fact, railcars are not considered in this model: trains are the smallest rolling stock elements.

Today, almost all modern rolling stocks are reversible and non-decomposable units: high-speed trains, recent regional trains, etc. But older trains still have a composition that varies depending on the destinations and weekdays; this implies different numbers and types of railcars, 1st vs 2nd class repartition... To keep the problem description simple, we don't handle this aspect here.

The successive stays of a given rolling stock unit in the system during the horizon are considered as different trains in this problem. For instance, a unit may be assigned to a departure $d \in \mathcal{D}$, leave the system and return back a few days later in it, associated with an arrival $a \in \mathcal{A}$: in this case, we consider two distinct trains. As described earlier, these trains might correspond to the same rolling stock unit, if $linkedDep_a = d$, but the trains $arrTrain_a$ and $depTrain_d$ are different because they correspond to distinct visits in the system. Once again, a train represents a *visit* of a rolling stock unit in the system, not the physical unit itself.

As a consequence, trains are considered only during a portion of the planning horizon: between their associated arrival (or the beginning of the horizon, if they are initially in the system) and the departure they are assigned to (or, if they are not assigned to any departure during the horizon, the end of the horizon).

A train $t \in \mathcal{T}$ belongs to a train category, $cat_t \in \mathcal{C}$, defining some common technical characteristics that t shares with other trains of the same category, $\mathcal{C}$ being the set of all train categories (see section 2.4.2).

2.4.1 Trains initially in the system

As mentionned earlier, some trains can be present in the system at the beginning of the planning horizon. In addition to its train categories, a train $t \in \mathcal{T}_I$ initially in the system is characterized by the following attributes:

- res_t: the resource used by t at d_1 $00:00:00$,
- $remDBM_t$: the remaining distance before maintenance of t,
- $remTBM_t$: the remaining time before maintenance of t.

2.4.2 Train categories

As introduced in Section 2.4, trains share common characterics defined by their belonging to some categories. In a sense, all trains belonging to the same category are identical, with only their initial maintenance conditions (remaining TBM and DBM) being different from one train to another.

In practice, trains are usually produced in series of a few dozen or hundred identical units which are eventually delivered by the manufacturers to the railways companies: these series typically constitute categories.

A train category $c \in \mathcal{C}$, where $\mathcal{C}$ is the set of all train categories, is defined by:

- a length: $length_c$,
- a compatibility group $catGroup_c$ expressing the physical and technical compatibility of trains with each other when two or more trains are assembled (see 2.5): two trains are compatible if and only if their respective compatibility groups are identical,
- a maximal distance before maintenance, $maxDBM_c$, expressed in km and defining the DBM of trains belonging to category c once they finish a maintenance operation of type "D"(restoring the DBM, see 2.6). This quantity represents the maximal number of km a train of category c can run between two maintenance operations of type "D".
- similarly, $maxTBM_c$ designates the maximal time before maintenance of trains of category c. This is the maximal time trains of category c can run between two maintenance operations of type "T".

- the time $maintTimeT_c$ required to perform a maintenance operation of type "T", and
- the time $maintTimeD_c$ required to perform a maintenance operation of type "D".

For the sake of simplicity, in the following we use the following convention (the characteristics of a train refer to those of its category). For any $c \in \mathcal{C}$ and any $t \in \mathcal{T}$ such that $cat_t = c$, we set:

$$\begin{aligned} length_t &= length_c \\ catGroup_t &= catGroup_c \\ maxDBM_t &= maxDBM_c \\ maxTBM_t &= maxTBM_c \\ maintTimeT_t &= maintTimeT_c \\ maintTimeD_t &= maintTimeD_c \end{aligned}$$

2.4.3 Preferred train reuses

From a practical point of view, the assignment of trains to departures is not always completely free. Indeed, in order to ensure the feasibility of the schedules, some decisions are planned in an earlier stage; train reuses are part of these decisions. They consist in pre-determining which arriving train is supposed to be assigned to a given departure.

This might be sub-optimal in some particular situations, especially when changes occur in the tactical planning and some assumptions do not hold anymore. However, this kind of information is shared among a high number of workers and these decisions are difficult to change because they induce disorganization in the operational management.

To represent this, some preferred train reuses are provided as an input in some instances (some instances might contain no preferred train reuse at all). A reuse $u \in \mathcal{U}$, where $\mathcal{U}$ denotes the set of all reuses, is defined by an arrival, arr_u, and a departure, dep_u. For such a reuse u, when possible, the train assigned to dep_u should be the same, ideally, as the one associated with arr_u. This might not be respected if this enables to be more efficient on other objectives, but in this case a non-satisfied reuse cost applies in the objective function.

It is assumed that any departure $d \in \mathcal{D}$ appears in at most one reuse, i.e. there is at most one reuse u such that $dep_u = d$. Conversely, for any $a \in \mathcal{A}$ there is at most one reuse u such that $arr_u = a$.

2.5 Joint-arrivals and joint-departures

When the number of expected passengers is high, some commercial trips are covered by more than one train: $n \geq 2$ trains are physically assembled to run together. We call a joint-arrival a combination of n simultaneous arrivals and a

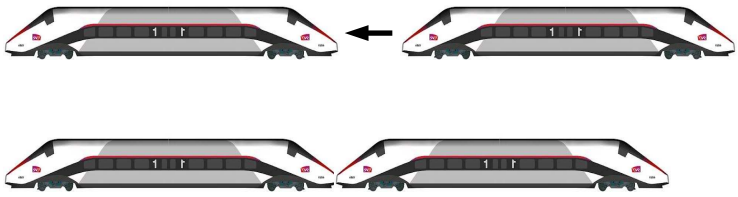

Figure 1: Junction of two trains

joint-departure the equivalent for n simultaneous departures, corresponding to n assembled trains arriving to or departing from the same plaftorm.

A joint-arrival concerns n arrivals, and expresses a synchronization of arrivals. A joint-departure consists of n departures, all having the same time; the associated trains must be assembled, on the same platform, in a certain order. Indeed, the roles played by the different trains differ according to the departure they are assigned to, and their order is important when they are disassembled, even when this happens outside of the considered system. Similarly, all the trains of a joint-arrival share the same time and the order of the arriving trains on the platform is an attribute of the joint-arrival.

Formally, a joint-departure $j \in \mathcal{J}_{dep}$, where $\mathcal{J}_{dep}$ is the set of all joint-departures, is defined by an ordered list of departures $jdList_j$. In a symmetrical way, $\mathcal{J}_{arr}$ represents the set of all joint-arrivals and any $j \in \mathcal{J}_{arr}$ is characterized by an ordered list of arrivals $jaList_j$.

The order of arrivals/departures is important because it enables to distinguish which train (associated with which departure/arrival) runs first, at the head of the "convoy". The convention adopted throughout this document is as follows: the first departure/arrival in the lists $jdList_j$ and $jaList_j$ is associated with the train located most on side A of the assigned platform (as explained later in Section 2.7, all resources have a side A and a side B), and the next departures/arrivals are associated with the next positions.

The junction of one train with another train, or with already assembled trains, is an operation which has a cost denoted $junCost$ and requires a duration denoted by $junTime$. This operation is presented in Figure 1. Starting from trains not assembled, the junction of n trains cannot be performed in a single junction operation: it requires $n-1$ junction operations and, consequently, costs $(n-1) \times junCost$ and requires a duration of at least $(n-1) \times junTime$.

Junction operations can be performed only on platforms, single tracks, maintenance facilities or yards (see types of resources in Section 2.7). Once assembled, the trains are considered as if they were a single train when they move on track groups: only one move is considered when assembled trains run on a track group. However, they are still considered as multiple trains on the other types of resources.

Symmetrically, a disjunction of trains has a cost ($disjCost$) and requires some time represented by $disjTime$. Like junctions, disjunctions can only be

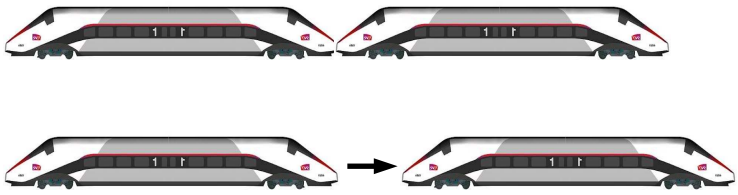

Figure 2: Disjunction of two trains

performed on platforms, single tracks, maintenance facilities or yards. A disjunction operation is shown in Figure 2.

The schedules associated with assembled trains should be the same between their junction and disjunction: beginning times on each resource as well as resources themselves must be identical as long as trains are assembled.

2.6 Maintenance

Trains must be maintained on a regular basis to be able to run in proper security and comfort conditions. Their ability to be assigned to departures is determined by comparing their DBM and TBM with the requirements of the departure (distance and time required for the journey following the departure).

The TBM is a value representing the time a train can run before a maintenance of type "T"needs to be performed. It is mostly associated with comfort considerations. The DBM is more related to security constraints. It represents the maximum distance a train can still run before the next maintenance of type "D".

For a given train performing no maintenance operation in the system, the DBM and TBM remain unchanged between an arrival and a departure: local moves between resources are neglected as their speed is slow and the associated distances are short with respect to those induced by departures; the time spent in the system by the train is not considered as affecting the TBM.

Two types of maintenance operations may be performed: maintenance of type "D"enables to restore the DBM of a train t to its maximum value, $maxDBM_t$, whereas maintenance of type "T"is its equivalent for time (restores TBM to its maximum value, $maxTBM_t$). A train may perform at most one operation of each type. Indeed, subsequent operations would have no effect on TBM/DBM because TBM/DBM would already be restored at their maximum value.

When an arrival a has a linked departure d (i.e. $linkedDep_a = d$), two cases must be distinguished. If d is not covered (i.e. if no train is assigned to d), then the remaining TBM and the remaining DBM of a are those provided in the input data for a. If d is covered, these characteristics are replaced by those induced by $depTrain_d$: the remaining TBM and DBM of a are deduced from those of the train assigned to d. If $d = linkedDep_a$, $t = depTrain_d$ and no

maintenance operation is performed on t, we have

$$remDBM_a = remDBM_t - reqDBM_d$$
$$remTBM_a = remTBM_t - reqTBM_d$$

If a maintenance operation of type "D"is performed on t, then

$$remDBM_a = maxDBM_t - reqDBM_d,$$

and if a maintenance operation of type "T"is performed on t,

$$remTBM_a = maxTBM_t - reqTBM_d.$$

For instance, if a train t has remaining DBM of 500 km and a remaining TBM of 48 hours, it can be assigned to a departure requiring a DBM of 450 km and a TBM of 24 hours. However, it may not be assigned to a departure requiring a DBM of 450 km and a TBM of 52 hours (this would violate the TBM requirement). It may also not be assigned to a departure requiring a DBM of 550 km and a TBM of 24 hours (this would then violate the DBM requirement).

Maintenance operations can only be performed on maintenance facilities (described in section 2.7.4) which are dedicated to only one type of maintenance. Consequently, a train may not perform both types of operations on the same maintenance facility. The duration of maintenance operations depends on the category of trains: for trains of category $c \in \mathcal{C}$, operations of type "D"last $maintTimeD_c$, and operations of type "T", $maintTimeT_c$.

Maintenance capacities in the system being limited, the number of maintenance operations which can be performed over a day in the system, i.e. over all maintenance facilities of the system, is bounded by a maximal value represented by $maxMaint$ (see constraint described in Section 4.3.7).

2.7 Infrastructure resources

Between arrivals and departures, trains are either moving or parking on tracks that we consider as resources.

In practice, trains can be very long and occupy more than only one track at a time, but we neglect this aspect here, we consider trains as if they were points which can instantly move from one resource to another.

Let $\mathcal{R}$ be the set of all resources. Resources can be either single tracks, platforms, maintenance facilities, yards or track groups; $\mathcal{S}$, $\mathcal{P}$, $\mathcal{F}$, $\mathcal{Y}$ and $\mathcal{K}$ represent respectively the set of all single tracks, all platforms, all maintenance facilities, all yards and all track goups of the system.

Single tracks, platforms and maintenance facilities represent portions of tracks considered in an individual manner, while track groups and yards are aggregated types of resources which usually contain more than only one track and contain switches to physically link the different tracks together. The next sections provide details for each of these types of resources. An example of resources configuration associated with a sample system are presented on Figure 3.

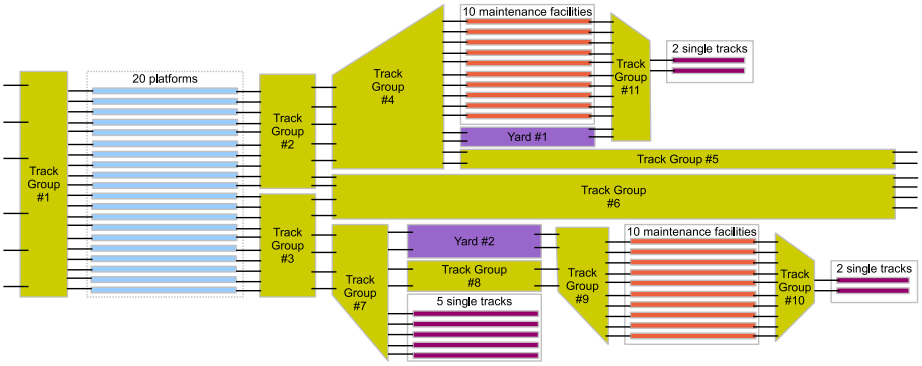

Figure 3: Example of infrastructure resources

2.7.1 Transitions between resources

A resource $r \in \mathcal{R}$ has a set $neighSet_r$ of neighbor resources, defining the possible transitions for trains, in a symmetrical way: if a resource r' is a neighbor of r, that is if it belongs to $neighSet_r$, r is a neighbor of r' and a train might use r' immediately after r (and vice-versa). On the contrary, if $r'' \notin neighSet_r$, the direct transition from r to r'' is not allowed (and neither from r'' to r): an intermediate resource must be used between r and r''. As we are interested only in transitions between different resources, a resource is not a neighbor of itself.

In our model, resources represent railways infrastructure elements which are in general *linear* and can be accessed from at most two sides, and in some cases from only one side. Given this aspect, the neighbors of a resource can then be divided into two subsets, one being physically associated with each "side" of the linear element. By convention, these two sides are denoted A and B. Hence, for any resource r, the set of neighbors of r is decomposed into two subsets:

$$neighSet_r = neighSet_r^A \cup neighSet_r^B$$

It is also assumed that a resource cannot be the neighbor of another resource both on A and B sides:

$$neighSet_r^A \cap neighSet_r^B = \varnothing$$

The transitions between a resource, r, and one of its neighbors, are performed through one of the entry/exit points, which we call *gates*, located on both sides of the resource. These gates correspond to the physical tracks linking the different resources. Let $\mathcal{G}_r$ denote the set of gates of resource r. A gate $g \in \mathcal{G}_r$ is defined by its side $side_g \in \{A, B\}$ and its index $ind_g \in \mathbb{N}^*$; r_g represents the resource g belongs to. The neighbor gate of g, $neigh_g$, is unique and represents the gate of a neighbor resource accessible through g (since a resource is not a neighbor of itself, $r_{neigh_g} \neq r$). The relation between a gate and its neighbor gate is reflexive: the neighbor gate of $neigh_g$ is g. The only exception concerns the

Figure 4: Track group representation example

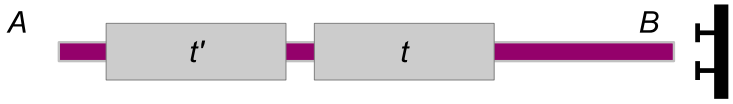

Figure 5: Example of resource with only one gate, on side A

gates at the boundaries of the system, which do not have a neighbor. Trains must run through these gates without neighbor to enter or exit the system.

On each side, the gates are ordered according to their physical positions. The indices of the gates follow this order. For instance, if a track group consists of tracks oriented following an East-West axis, the gates on each side are ordered from North to South. On side A, the gate most on the North is called A1, the next gate A2, and so on, until An, where n is the number of gates on side A. For gate A1, A represents the side of the gate, 1 its index. Figure 4 presents an exemple for a track group. Associated with each gate of a particular resource is exactly one gate associated with one of its neighbors (or no neighbor if the gate represents is at the boundary of the system).

Single tracks, platforms and facilities, representing individual tracks, have at most one gate on side A, and at most one gate on side B (hence, at most one neighbor resource on each side):

$$\forall r \in \mathcal{S} \cup \mathcal{P} \cup \mathcal{F}, |neighSet_r^A| \leq 1 \text{ and } |neighSet_r^B| \leq 1.$$

When only one side is accessible, for instance a platform in a terminal station where the track ends, we use the convention that only side A is accessible: $neighSet_r^A \neq \varnothing$ and $neighSet_r^B = \varnothing$.

A resource $r \in \mathcal{S} \cup \mathcal{P} \cup \mathcal{F}$ with only one gate on side A and no gate on side B can be seen as a stack. It must be managed in a Last In First Out (LIFO) way. This means that a train t cannot leave r at $h \in \mathcal{H}$ if another train t' has come later on r and has not yet left r. A resource $r' \in \mathcal{S} \cup \mathcal{P} \cup \mathcal{F}$ with one gate on side A and one gate on side B can be seen as a double-ended queue. A train t cannot leave r' through the gate on side A if another train t' has come later through the gate of side A. This characteristic is represented on Figure 5. Associated constraints are presented in Section 4.3.8.

Only track groups and yards may have more than one gate on each side. For these types of resources, two adjacent resources might be linked by more

than one gate: the gate used for the transition between two resources has to be specified.

Some resources can be used only by some particular train categories. The set of train categories compatible with resource r is denoted by $compCatRes_r$. For instance, some infrastructure resources might not be electrified, which prevents their use by electrical rolling stock; in general, maintenance facilities are dedicated to some categories only.

To take into account that some operations must be performed on a train to change its direction (in particular, drivers must walk to go to the other extremity of the train), a train entering on a single track through a given side must stay a minimum amount of time on it before going back through this same side (see constraint in Section 4.1.8). This time is represented by $revTime$.

2.7.2 Single tracks

Some tracks of the system, which are not located in stations (i.e. which do not allow boarding and unboarding of passengers and, hence, cannot be used for arrivals or departures), are considered individually, not part of yards or track groups. They are called *single tracks*. $\mathcal{S} \subset \mathcal{R}$ is the set of single tracks.

A single track resource $s \in \mathcal{S}$ has a length $length_s$ and a capacity $capa_s$. At any time $h \in \mathcal{H}$, both the total length of trains and the number of trains parked on s must not exceed the length of the track and its capacity (see constraints in Section 4.3.2 and 4.3.6).

The order of trains on single tracks must be consistent with their moves. As trains cannot fly over each other (things would be much easier if they could!), they must respect the order of their arrival on the track to leave it (see Section 4.3.8).

2.7.3 Platforms

In our model, platforms represent tracks within the train station where passengers can board and unboard the trains. They are very similar to single tracks in the sense that the order of trains must be consistent with their respective times of moves. However, they do not have a capacity expressed in maximal number of trains. Moreover only platforms can be assigned to arrivals and departures. $\mathcal{P} \subset \mathcal{R}$ denotes the set of all platforms. Each platform $p \in \mathcal{P}$ has a length $length_p$.

If the duration of use of a platform is too short before a departure $d \in \mathcal{D}$, less than $idealDwell_d$, passengers might not be able to board the train in a comfortable way. This is penalized by a cost. Symmetrically, trains staying more than $idealDwell_d$ on platforms before departure are penalized because platforms are considered as critical resources (see Section 5.3.3). In any case, trains may not use a platform for more than $maxDwell_d$ before departure (see constraint in Section 4.3.3).

Similar considerations apply to arriving trains, which should stay on platforms a duration close to $idealDwell_a$, and in any case less than $maxDwell_a$.

2.7.4 Maintenance facilities

Maintenance facility resources are special tracks inside maintenance workshops. They are used to periodically reset the DBM and TBM of trains. The set of maintenance facilities is denoted by $\mathcal{F} \subset \mathcal{R}$.

A maintenance facility $f \in \mathcal{F}$ is characterized by a type $type_f \in \{\text{“}D''\text{, “}T''\}$ indicating which type of operations can be performed. If $type_f$ equals “D”, only operations of type “D”can be performed; otherwise, if $type_f$ equals “T'', only operations of type “T”can be performed. As a consequence, either the DBM or the TBM is restored by a maintenance facility resource, in an exclusive manner. It is also characterized by a length $length_f$ which may not be exceeded by the total length of trains using it.

2.7.5 Track groups

Track groups are sets of tracks used by trains to move throughout the system.

A track group represents a sub-part of the rail network in the considered system. Its real physical configuration in terms of tracks and switches linking them can be very complex; we don't consider this complexity here, we rather see it as a black box with some indications on how to identify conflicts.

The set of all track groups is represented by $\mathcal{K} \subset \mathcal{R}$. A track group $k \in \mathcal{K}$ is supposed to be used for train moves: the duration of use of k by any train (i.e. travel time) is a constant denoted by $trTime_k$. It is the time required by a train to enter the track group k on one side and exit at the opposite side. Indeed, a train entering on one side of a track group must exit on the other side. All gates of one side are reachable from all gates of the opposite side. As a consequence, if a track group has n gates on one side and m gates on the other side, then $n.m$ different paths are possible in each direction. Besides, $hwTime_k$ represents the headway of the track group: this is a security time which must be respected at any place between two trains.

Figure 4 shows an example of track group representation with n gates on side A and m gates on side B.

When several trains use a track group over the same time period, conflicts might occur between them. A conflict is an unwished situation where two running trains might come too close to each other, and one has to stop one of them to respect security distances. Once again, we adopt here a “black-box” approach where the full complexity of the track group is eluded. Real conflicts should be identified through a very detailed description of all infrastructure equipments and signaling systems; here, conflicts model non-robustness, that is trains are likely to stop due to headway constraints.

The way conflicts are detected is exposed in Section 5.2.1.

2.7.6 Yards

A yard is also a set of tracks, but it is mainly used for parking. Therefore, contrary to track groups which are dedicated to train moves, the duration of

use by trains is not fixed, trains can stay on yards with no time restriction. The set of yards is denoted by $\mathcal{Y} \subset \mathcal{R}$.

A yard $y \in \mathcal{Y}$ has a limited physical capacity on the number of trains which can be handled simultaneously, $capa_y$, which is a way to model the number of tracks and the maximal number of trains that can be parked simultaneously. To reflect this fact, any train using the yard beyond its capacity is penalized by a cost in the objective function (see Section 5.2.2).

This capacity is an approximation because in practice, the maximal number of train able to dwell simultaneously on a yard depends on their type, the number and the length of the tracks inside the yard, the position of switches... So, similarly to conflicts on track groups, going beyond $capa_y$ is not desirable because it is not robust, but this does not necessarily mean that the situation is infeasible from a practical point of view.

2.7.7 Initial train location

It is assumed that trains initially in the system are not traveling on track groups at h_0 (in other words, for any $t \in \mathcal{T}_I$, $res_t \notin \mathcal{K}$). Moreover, we assume that when several trains are initially in the system at the beginning of the horizon on the same resource of type single track, platform or maintenance facility, they are positionned by order of appearance in the data input file, starting from side A (the first train appearing in the file is the most on side A, the next ones are on the next positions towards side B).

2.7.8 Imposed resource consumptions

Some aspects of the system we consider are external to our decision perimeter. They are considered fixed and cannot be changed. For instance, some trains use resources during the horizon whereas they are not part of $\mathcal{T}$: infrastructure maintenance trains, trains not terminating at the train station and continuing their journey, or trains from other companies, on which no decision is to be made. They should not be considered the same way in the sense that no decision should be made for them. However they do use the same resources as those of $\mathcal{T}$, i.e. resources of $\mathcal{R}$. Moreover, some resources might be unavailable due to opening times of resources or infrastructure maintenance works. To represent this, pre-determined consumptions of resources are imposed over the horizon on the different resources. Depending on the type of resource, these imposed consumptions have different characteristics.

The set of imposed resource consumptions is represented by $\mathcal{I}$. An imposed consumption $i \in \mathcal{I}$ refers to an associated resource, denoted res_i. If res_i is an individual track (single track, platform or maintenance facility), it is considered unavailable between a beginning time, beg_i, and an end time, end_i. No train is allowed to use res_i during the interval $[beg_i, end_i[$.

If res_i is a yard ($y \in \mathcal{Y}$), then i is defined by a number of trains, nb_i, using the yard y between beg_i and end_i. It is equivalent to a temporary reduction

of capacity of the yard by nb_i units. We assume that two distinct imposed consumptions for the same yard do not overlap.

Finally, if res_i is a track group $k \in \mathcal{K}$, i represents the move of a train over the track group; it is then defined by an origin gate, o_i, a destination gate, d_i, and the time the train enters the track group at o_i, h_i. These imposed train paths over track groups are considered exactly the same way as for trains in $\mathcal{T}$ when detecting conflicts.

3 Solution representation

A solution to the problem is composed of a set of schedules, each denoted by $sched_t$, one for each train $t \in \mathcal{T}$. The schedule $sched_t$ of train t is a sequence of events during its presence in the system, along with details such as the time of each event, the resources used, etc. With this information for every train, it is possible to derive the status of the system and each of its resources at any time during the horizon.

The considered events concerning a train are its arrival and departure, its entrance in and exit from the system or any resource of the system, and the beginning and end of junction, disjunction and maintenance operations. These types of events are respectively denoted by `Arrival`, `Departure`, `EnterSystem`, `ExitSystem`, `EnterResource`, `ExitResource`, `BegJunction`, `EndJunction`, `BegDisjunction`, `EndDisjunction`, `BegMaintenance` and `EndMaintenance`.

With any event $e \in \mathcal{E}$, where $\mathcal{E}$ represents the set of all events in the solution, is associated a train, t_e, a time, h_e, an event type y_e, a resource, r_e, a gate on r_e, g_e, and a complement, c_e. Depending on the type of event, some of these characteristics may not be relevant (a detailed description of events is provided in Section 6.2.2, it defines for each type of event the expected characteristics).

Examples of schedules associated with trains departing together in a joint-departure are provided in Tables 4 to 6. Finally, the constraints defining the feasibility of a solution are presented in Section 4.1, and the evaluation of solutions (objective functions) is precised in Section 5.

4 Constraints

A solution is feasible if and only if all constraints described in this section are satisfied. While most of them were introduced earlier in the previous sections, along with the concepts, the following sections describe in a more formal way the constraints of the problem.

4.1 Schedule properties

4.1.1 Entrance in the system

The schedule of any train $t \in \mathcal{T}$ must begin with an `EnterSystem` event.

If t is initially in the system at the beginning of the horizon ($t \in \mathcal{T}$), the `EnterSystem` event must occur at d_1 00 : 00 : 00 and be immediately followed by an `EnterResource` event on its initial resource, res_t, also at d_1 00 : 00 : 00.

If t is associated with an arrival $a \in \mathcal{A}$ ($t \in \mathcal{A}$), the `EnterSystem` event must be followed by a sequence of `EnterResource`/`ExitResource` events on the track groups imposed by the arrival sequence $arrSeq_a$, in the imposed order. Then, an `EnterResource` on the platform assigned to a must occur at $arrTime_a$.

4.1.2 Exit of the system

The schedule of any train $t \in \mathcal{T}$ must end with an `ExitSystem` event.

If t is not assigned to any departure, then the `ExitSystem` occurs at d_{nbDays} 23 : 59 : 59, immediately preceded by an `ExitResource` event, also at d_{nbDays} 23 : 59 : 59, on the last occupied resource.

If t is assigned to departure $d \in \mathcal{D}$, the `ExitResource` event must be preceded by a sequence of `EnterResource`/`ExitResource` events on the track groups imposed by the departure sequence of d, $depSeq_d$, in the imposed order. This departure sequence itself must be preceded by an `ExitResource` event on the platform assigned to d at $depTime_d$.

4.1.3 Use of a resource

Every `EnterResource` event e in a schedule $sched_t$, for any $t \in \mathcal{T}$, must be followed by an `ExitResource` event e', concerning the same resource (i.e. such that $r_e = r_{e'}$). Only events of type `Arrival`, `Departure`, `Beg`/`EndJunction`, `Beg`/`EndDisjunction` and `Beg`/`EndMaintenance` may occur between e and e', and the resource associated with these events must be r_e.

4.1.4 Transition from a resource to a neighbor resource

Every `ExitResource` event e in a schedule $sched_t$, for any $t \in \mathcal{T}$, must be immediately followed by an `ExitSystem` or an `EnterResource` event e', concerning a different resource (i.e. such that $r_e \neq r_{e'}$). Moreover, the entry gate $g_{e'}$ on e' has to be the neighbor of the exit gate g_e on e ($neigh_{g_e} = g_{e'}$), and the times of both events must be the same ($h_e = h_{e'}$).

4.1.5 Maintenance operations

The schedule of any train $t \in \mathcal{T}$ may or may not contain `BegMaintenance` events. If it contains such a `BegMaintenance` event e, then e must be between an `EnterResource` and an `ExitResource` event on a resource f of type maintenance facility ($f \in \mathcal{F}$), and the type of maintenance has to be consistent with the type of facility (if $type_f =$"D", then $c_e =$ "D"). It must also be immediately followed by an `EndMaintenance` event, e', such that:

$$c_e = \text{“D”} \implies h_{e'} = h_e + maintTimeD_t. \tag{1}$$

$$c_e = \text{“T”} \implies h_{e'} = h_e + maintTimeT_t. \tag{2}$$

A train t may not perform more than one maintenance operations of each type (i.e. $sched_t$ contains at most one operation of type "D"and one operation of type "T").

4.1.6 Junction and disjunction operations

Likewise, the schedule of any train $t \in \mathcal{T}$ may or may not contain `BegJunction` and `BegDisjunction` events. If it contains a `BegJunction` or a `BegDisjunction` event e, then e must be between an `EnterResource` and an `ExitResource` event on a resource r of type maintenance facility, platform, single track or yard. It must also be immediately followed by an `EndJunction` (if e is `BegJunction`) or an `EndDisjunction` (if e is `BegDisjunction`) event, e', such that:

$$y_e = \texttt{BegJunction} \implies h_{e'} = h_e + junTime. \tag{3}$$

$$y_e = \texttt{BegDisjunction} \implies h_{e'} = h_e + disjTime. \tag{4}$$

Note that a junction or a disjunction operation may assemble or disassemble only *two* trains or sets of trains at a time. Moreover, only trains on adjacent positions on a track may perform a junction operation; this operation is not possible if other trains are located between the trains to be assembled.

4.1.7 Duration of use of track groups

In the schedule of any train $t \in \mathcal{T}$, any `EnterResource` event e on a track group must be immediately followed by an `ExitResource` event e' that occurs after a duration $trTime_k$, and on the opposite side:

$$h_{e'} = h_e + trTime_k, \tag{5}$$

$$side_{g_{e'}} \neq side_{g_e}. \tag{6}$$

4.1.8 Minimum reverse time

In the schedule of any train $t \in \mathcal{T}$, any `EnterResource` event e on a resource of type single track, platform, maintenance facility or yard, followed by an `ExitResource` event e' on the same resource and on the same side, must be such that:

$$side_{g_{e'}} = side_{g_e} \implies h_{e'} \geq h_e + revTime. \tag{7}$$

4.2 Assignments

4.2.1 At most one train assigned per departure

For any departure $d \in \mathcal{D}$, there must not be two different trains with a departure event associated with d. If d is not part of a joint-departure, the assigned train, if any, must not be assembled with another train at the time of departure (in other words, two trains may not leave a platform assembled for a departure except for a joint-departure).

4.2.2 Required DBM for a departure

For any train $t \in \mathcal{T}$ and any departure $d \in \mathcal{D}$, assigning t to d implies that its remaining DBM is greater than or equal to the required DBM of d.

Two cases are distinguished. If t performs a maintenance operation of type "D", then its DBM at departure time is equal to $maxDBM_t$, hence:

$$depTrain_d = t \implies maxDBM_t \geq reqDBM_d. \tag{8}$$

If t does not perform a maintenance operation of type "D", then its DBM is unchanged:

$$depTrain_d = t \implies remDBM_t \geq reqDBM_d. \tag{9}$$

In both cases, its DBM has to be sufficient for d.

Recall that if an arrival $a \in \mathcal{A}$ has a linked departure $d' = linkedDep_a$ and if a train t' is assigned d', the remaining DBM and TBM of the train t'' associated with a depend on t':

$$remDBM_{t''} = remDBM_{t'} - reqDBM_{d'}$$
$$remTBM_{t''} = remTBM_{t'} - reqTBM_{d'}$$

4.2.3 Required TBM for a departure

Similarly, for any train $t \in \mathcal{T}$ and any departure $d \in \mathcal{D}$, assigning t to d implies that its TBM is greater than or equal to the required TBM of d.

If t performs a maintenance operation of type "T", then its TBM at departure time is equal to $maxTBM_t$, hence:

$$depTrain_d = t \implies maxTBM_t \geq reqTBM_d. \tag{10}$$

If t does not perform a maintenance operation of type "T", then its TBM is equal to $remTBM_t$:

$$depTrain_d = t \implies remTBM_t \geq reqTBM_d. \tag{11}$$

4.2.4 Departure - train category compatibility

The categories of trains must be compatible with the departures they are assigned to:

$$\forall d \in \mathcal{D}, \forall t \in \mathcal{T}, depTrain_d = t \implies cat_t \in compCatDep_d. \tag{12}$$

4.2.5 Assembled train category compatibility

When two trains are assigned to departures belonging to a joint-departure, the categories of the assigned trains must be compatible with each other:

$$\forall j \in \mathcal{J}_{dep}, \forall (d, d') \in jdList_j, \forall (t, t') \in \mathcal{T} \text{ such that } t \neq t',$$
$$depTrain_d = t \text{ and } depTrain_{d'} = t' \implies catGroup_t = catGroup_{t'} \tag{13}$$

4.3 Resource usage

At the time a train exits a resource r and enters the next one in its schedule, r', it is considered only using r', not r.

In the following, for any train $t \in \mathcal{T}$, any time $h \in \mathcal{H}$ and any resource $r \in \mathcal{R}$, we use the notation $use_{t,h} = r$ to denote the resource r used by t at h. If t is in the system at h, $use_{t,h} = r$ if and only if the last `EnterResource` event in the schedule of t before h or at h is associated with r. Otherwise, (if h is before its entrance in or after its exit from the system), we set $use_{t,h} = \varnothing$.

For any resource r of type single track or yard, $nbTrain_{r,h}$ designates the number of trains on r at h.

When r is a single track, we have:

$$\forall h \in \mathcal{H}, \forall r \in \mathcal{S}, nbTrain_{r,h} = |\{t \in \mathcal{T}; use_{t,h} = r\}|. \tag{14}$$

When r is a yard, $nbTrain_{r,h}$ includes imposed consumptions. Then, we have:

$$\forall h \in \mathcal{H}, \forall r \in \mathcal{Y}, nbTrain_{r,h} = |\{t \in \mathcal{T}; use_{t,h} = r\}| + \sum_{\substack{i \in \mathcal{I} \\ res_i = r \\ h \in [beg_i, end_i[}} nb_i. \tag{15}$$

4.3.1 Resource - train category compatibility

For any train $t \in \mathcal{T}$, for any resource $r \in \mathcal{R}$, if t uses r during its schedule, r must be compatible with the category of t:

$$\forall t \in \mathcal{T}, \forall r \in \mathcal{R}, \\ \exists h \in \mathcal{H} \text{ such that } use_{t,h} = r \implies cat_t \in compCatRes_r. \tag{16}$$

4.3.2 Single track capacity

At any time, the number of trains using a single track $s \in \mathcal{S}$ may not exceed its capacity:

$$\forall s \in \mathcal{S}, \forall h \in \mathcal{H}, nbTrain_{s,h} \leq capa_s. \tag{17}$$

4.3.3 Maximum duration of use of a platform

A platform may be temporarily used for purposes other than an arrival or a departure (e.g. in some configurations, trains have to run through a train station to reach another station). This is allowed but in such cases (use of platform in absence of any `Arrival` or `Departure`), the duration of use of the platforms must not exceed a constant duration denoted by $maxDwellTime$.

When a platform is used for an arrival $a \in \mathcal{A}$, the stay of the associated train may not exceed $maxDwell_a$. Likewise, when a platform is used for a departure $d \in \mathcal{D}$, the assigned train may not stay more than $maxDwell_d$. Finally, if a platform is used for an arrival a iemmediately followed by a departure d, the duration of the stay may not exceed $\max(maxDwell_a, maxDwell_d)$.

4.3.4 Minimum duration of use of a resource

For any train t and any resource r used by t (except if r is a track group), the time difference between the associated `EnterResource` and `ExitResource` events must be greater than or equal to a constant duration represented by $minResTime$.

4.3.5 Imposed consumption

For any imposed consumption $i \in \mathcal{I}$ such that the resource r_i is either a single track, a platform or a maintenance facility, no train should use res_i between beg_i and end_i:

$$\forall i \in \mathcal{I}, res_i \in \mathcal{S} \cup \mathcal{P} \cup \mathcal{F} \implies \forall h \in [beg_i, end_i[, \forall t \in \mathcal{T}, use_{t,h} \neq res_i. \quad (18)$$

4.3.6 Track length

At any time, the total length of trains using a resource of type single track, maintenance facility or platform, must be less than or equal to the length of the resource. The total length of trains using a resource is defined as the sum of their respective lengths:

$$\forall r \in \mathcal{S} \cup \mathcal{F} \cup \mathcal{P}, \forall h \in \mathcal{H}, \sum_{\substack{t \in \mathcal{T} \\ use_{t,h}=r}} length_t \leq length_r. \quad (19)$$

4.3.7 Capacity of maintenance resources

For each day i between 1 and $nbDays$, at most $maxMaint$ maintenance operations may start. Let $maintD_{t,i}$ (respectively $maintT_{t,i}$) be equal to 1 if and only if a `BegMaintenance` event of type "D"(respectively, type "T") occurs in the schedule of t on day i (i.e. during the time window $[d_i\ 00:00:00; d_i\ 23:59:59]$), 0 otherwise. Then, the following constraint must hold:

$$\forall i \in [\![1, nbDays]\!], \sum_{t \in \mathcal{T}} maintD_{t,i} + maintT_{t,i} \leq maxMaint. \quad (20)$$

4.3.8 Train order on individual tracks

Depending on the presence of neighbors on both sides (A and B), single tracks, platforms and maintenance facilities are handled as stacks or as double-ended queues.

For any two trains t_1 and t_2, not assembled and using a common resource r of type single track, platform or maintenance facility, let beg_i and end_i be respectively the times t_i enters and, respectively, exits r, and let respectively $from_i$ and to_i be the side of the gate through which t_i enters and, respectively, exits r. Without loss of generality, let us assume that t_1 enters r before t_2 (i.e. $beg_1 < beg_2$).

Then, depending on the sides of enter/exit of t_1 and t_2, the constraints listed in Table 1 apply.

$from_1$	to_1	$from_2$	to_2	Constraint
A	A	A	A	$end_2 < end_1$ or $beg_2 > end_1$
A	A	A	B	$beg_2 > end_1$
A	A	B	A	$end_2 > end_1$
A	A	B	B	$\varnothing$ (no constraint)
A	B	A	A	$\varnothing$ (no constraint)
A	B	A	B	$end_2 > end_1$
A	B	B	A	$beg_2 > end_1$
A	B	B	B	$beg_2 > end_1$ or $end_2 < end_1$
B	A	A	A	$beg_2 > end_1$ or $end_2 < end_1$
B	A	A	B	$beg_2 > end_1$
B	A	B	A	$end_2 > end_1$
B	A	B	B	$\varnothing$ (no constraint)
B	B	A	A	$\varnothing$ (no constraint)
B	B	A	B	$end_2 > end_1$
B	B	B	A	$beg_2 > end_1$
B	B	B	B	$end_2 < end_1$ or $beg_2 > end_1$

Table 1: Constraints on train order on individual tracks

$from_1$	$from_2$	Relative entrance time	Resulting train order
A	A	$beg_1 < beg_2$	t_2, t_1
A	A	$beg_1 > beg_2$	t_1, t_2
A	B		t_1, t_2
B	A		t_2, t_1
B	B	$beg_1 < beg_2$	t_1, t_2
B	B	$beg_1 > beg_2$	t_2, t_1

Table 2: Train order after junction operations

4.3.9 Train order after junction operations

On any resource r where junction is allowed (i.e. facilities, platforms, single tracks and yards), the order of trains after a junction operation is derived from their respective times and sides of entrance.

Let t_1 and t_2 be two trains performing a junction operation on r. Using the same notations as in 4.3.8 and following the convention adopted earlier, stating that the order of trains on a particular resource r is expressed starting from side A of r, Table 2 presents the resulting order of trains after the junction operation.

Note that despite their macroscopic representation, yards are also subject to this constraint because trains performing a junction must be located on the same physical track within the yard (note that these internal tracks are not considered as resources of the problem, so this is implicit). This is illustrated by an example on Figure 6 where trains t_1 and t_2 are assembled after a junction. If other trains are not assembled or dissembled with t_1 or t_2, the physical track

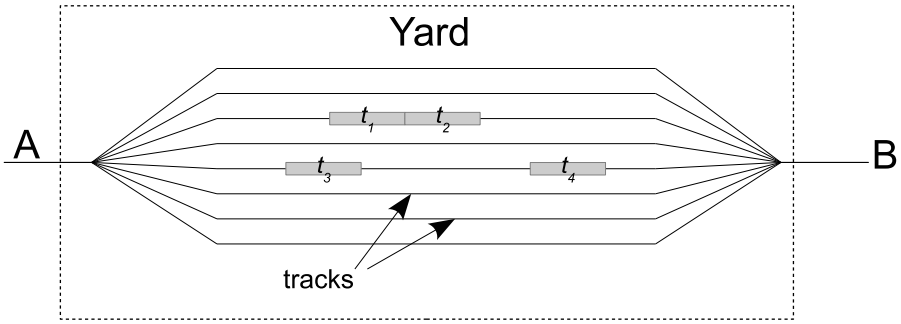

Figure 6: Example of yard where trains t_1 and t_2 perform a junction

to_1	to_2	Train order before disjunction	Constraint
A	A	t_2, t_1	$end_1 > end_2$
A	A	t_1, t_2	$end_1 < end_2$
A	B	t_1, t_2	$\varnothing$ (no constraint)
A	B	t_2, t_1	infeasible
B	A	t_1, t_2	infeasible
B	A	t_2, t_1	$\varnothing$ (no constraint)
B	B	t_1, t_2	$end_1 > end_2$
B	B	t_2, t_1	$end_1 < end_2$

Table 3: Constraint on exit after disjunction operations

they are using is not known and in this case we assume that they do not block moves. In short, the relative positions of trains on a yard only has a sense for trains being assembled or disassembled on this yard.

4.3.10 Exit from resource after disjunction operations

In a symmetrical way, the order of trains after a disjunction operation on a resource r implies some constraints on their exit times and sides of r. These constraints are expressed in Table 3. Likewise, this also applies to yards.

4.4 Assembled trains

4.4.1 Simultaneous events of assembled trains

When trains are assembled, they must have the same events in their associated schedules during their whole "assembled period". That is events have same time, same type, same resource, same gate; only concerned train differs for each event. Events must also have the same complement except for the `BegJunction` and `EndJunction` events at the beginning and end of the assembled period, if any (see description of `BegJunction` and `EndJunction` in Section 6.2.2).

This assembled period starts from their common entrance in the system (joint-arrival) or the time they are assembled (junction operation) until their common exit from the system (joint-departure), or until a disjunction operation occurs in their schedules.

4.4.2 Joint-departure: departures order

When $n \geq 2$ trains are assigned to trains belonging to a joint-departure $j \in \mathcal{J}_{dep}$, these trains must be assembled at least $minAsbTime$ before departure time and their respective positions must respect the order imposed by the joint-departure, $jdList_j$.

In particular, the junction operation(s) for a joint-departure, if any, must be performed in a way such that the respective positions of trains on the departure platform are coherent with the joint-departure order. The positions of trains after a junction operation are given by the respective entrance times and sides of the trains being assembled (see 4.3.9 and 4.3.10).

At every transition of assembled trains from resource r to a neighbor resource r', let s be the side of the gate through which the assembled trains leave r and s' the side of the gate through which the assembled trains enter r'. Then, the respective positions of trains after the transition is *unchanged* (i.e. trains on side A of r remain on side A of r' after transition) if $s \neq s'$, and *inverted* (i.e. trains on side A of r arrive on side B of r' after transition) if $s = s'$.

If trains assembled at their entrance in the system for a joint-arrival are assigned to a joint-departure without junction operations, the successive transitions over the resources of the system must also respect the order of $jdList_j$.

Finally, if one of the departures of a joint-departure $j \in \mathcal{J}_{dep}$ has no train assigned, the trains assigned to other departures of j must still respect the order of $jdList_j$. Formally, if d is before d' in $jdList_j$, then the train $depTrain_d$ must be located on side A of the departure platform with respect to $depTrain_{d'}$ (according to convention on the order of joint-departures defined in Section 2.5).

4.4.3 Joint-arrival: arrival order

In a symmetrical way, after a joint-arrival $j \in \mathcal{J}_{arr}$, the moves of the associated trains after a disjunction operation must respect the order of trains induced by $jaList_j$ and the potential transitions on the different resources performed while trains are assembled.

5 Objectives

The objective function f is used to evaluate the quality of feasible solutions. It is divided into three objectives f_1, f_2 and f_3, to be minimized, ordered by descending significance:

1. Number of uncovered departures (f_1)
2. Number of conflicts on track groups and yard overload (f_2)

3. Performance costs (f_3)

The ranking of solutions is based on a multi-objective lexicographic order: a solution s is considered better than s' if and only if, for some $i \in [1,3]$, $f_i(s) < f_i(s')$ and for all j such that $j < i, j \geq 1, f_j(s) = f_j(s')$.

5.1 Uncovered departures

If a departure has no train assigned to it, it is considered as *uncovered*. Covering all departures is the most important criterion. The number of uncovered departures is given by:

$$f_1(s) = |\{d \in \mathcal{D}; depTrain_d = \varnothing\}| \tag{21}$$

If no train is assigned to a departure d belonging to a joint-departure j, d is considered uncovered. If trains are assigned to the other departures of j, these departures are covered but their order has to be coherent with the order defined by the joint-departure (see constraint is Section 4.4.2).

5.2 Conflicts on track groups and yard overloads

As discussed earlier, conflicts on track groups and yard overloads are undesirable because they induce a risk of infeasibility of the solution in practice. Note that this is only a risk, because our model is macroscopic. Once again, the only way to precisely detect a real infeasibility (i.e. impossibility to run through the track groups or to enter a yard already saturated) would be to consider all the details of tracks and switches along with the signaling systems, which is far beyond the scope of our model. This is why we do not explicitly handle conflicts on track groups and yard overloads as constraints but rather as a significant criterion in the objective function with macrosopic rules to detect potential issues.

5.2.1 Conflicts on track groups

Depending on their respective times, conflicts may occur between two moves m_1 and m_2 on a track group. Moves m_1 and m_2 can be associated with consecutive `EnterResource` and `ExitResource` events in the schedule of trains of $\mathcal{T}$, or imposed consumptions on a track group. Let $\mathcal{M}$ represent the set of all moves on track groups.

For any move m_i on track group k, let o_i, d_i and h_i respectively denote its origin gate (i.e. the gate through which m_i enters k), its destination gate (i.e. through which m_i exits k) and the time it enters the track group at o_i.

It is recalled that $trTime_k$ represents the travel time of a train on k, $hwTime_k$ the headway time on k, meaning the minimum buffer time between two trains to respect security distances, and that assembled trains entering a track group count for only one move.

Let m_1 and m_2 be two moves on the same track group. We consider that a conflict occurs in the following cases:

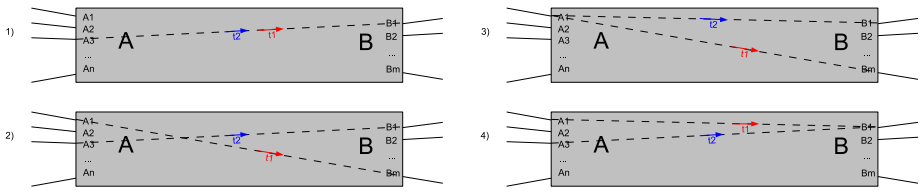

Figure 7: Track group conflict: example configurations for case 1

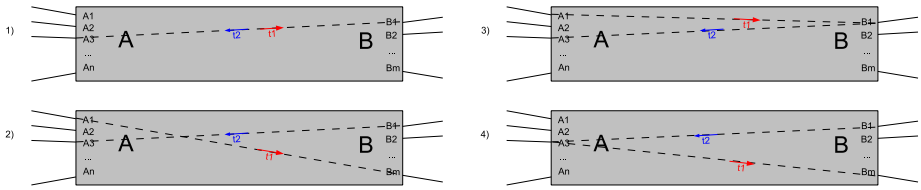

Figure 8: Track group conflict: example configurations for case 2

- Case 1: m_1 and m_2 are on intersecting paths, in the same direction, with insufficient buffer time (see Figure 7 for examples of configurations where conflicts might arise between two trains in the same direction).
  - the indices of destination gates of m_1 and m_2 are equal, or inverted with respect to those of origin gates: $(ind_{o_1} - ind_{o_2}) \times (ind_{d_1} - ind_{d_2}) \leq 0$,
  - m_1 and m_2 are in the same direction ($side_{o_1} = side_{o_2}$), and
  - the headway time may not be respected ($|h_1 - h_2| < hwTime_k$).
- Case 2: m_1 and m_2 are on intersecting paths, in opposite directions, with insufficient buffer time (see Figure 8 for some potential conflicts that might occur for case 2).
  - the indices of the gates used by m_1 and m_2 on each side are equal, or inverted with respect to those used on the other side: $(ind_{o_1} - ind_{d_2}) \times (ind_{d_1} - ind_{o_2}) \leq 0$,
  - m_1 and m_2 are in opposite directions ($side_{o_1} \neq side_{o_2}$), and
  - the headway time may not be respected ($|h_1 - h_2| < trTime_k + hwTime_k$).

With this definition, the number of conflicts is given by:

$$\begin{aligned} &nbConflicts = \\ &|\{(m_1, m_2, k) \in \mathcal{M}^2 \times \mathcal{K}; \\ &\qquad (ind_{o_1} - ind_{o_2}) \times (ind_{d_1} - ind_{d_2}) \leq 0, side_{o_1} = side_{o_2}, \\ &\qquad |h_1 - h_2| < hwTime_k\}| \\ &+|\{(m_1, m_2, k) \in \mathcal{M}^2 \times \mathcal{K}; \\ &\qquad (ind_{o_1} - ind_{d_2}) \times (ind_{d_1} - ind_{o_2}) \leq 0, side_{o_1} \neq side_{o_2}, \\ &\qquad |h_1 - h_2| < trTime_k + hwTime_k\}| \end{aligned} \tag{22}$$

5.2.2 Yard overload

For every entrance of a train on a yard $y \in \mathcal{Y}$ (i.e. for any event e such that $r_e = y$ and $y_e =$ `EnterResource`), a cost applies if the number of trains using yard y at h_e is strictly greater than its capacity, $capa_y$.

The same principle applies for an imposed consumption $i \in \mathcal{I}$ starting on y at a moment where the number of trains using y is greater than its capacity.

The number of occurences of yard overload in then defined by:

$$\begin{aligned} &nbYardOverload = \\ &\quad |\{(e, y) \in \mathcal{E} \times \mathcal{Y}; y_e = \texttt{EnterResource}, r_e = y, nbTrain_{y,h_e} > capa_y\}| \\ &\qquad + |\{(i, y) \in \mathcal{I}; res_i = y, nbTrain_{y,beg_i} > capa_y\}|. \end{aligned} \tag{23}$$

Finally, we have the following definition of f_2:

$$f_2(s) = nbConflicts + nbYardOverload \tag{24}$$

5.3 Performance cost

Performance cost constitutes the third criterion. It is composed of a weighted sum of individual costs which are:

- Over-maintenance cost (f_3^{over}),
- Train junction and disjunction operation costs (f_3^{jun}),
- Platform usage costs (f_3^{plat}),
- Non-satisfied preferred platform assignment cost (f_3^{pref}), and
- Non-satisfied train reuse cost (f_3^{reuse}).

f_3 is decomposed this way:

$$f_3 = f_3^{maint} + f_3^{pref} + f_3^{plat} + f_3^{jun} + f_3^{reuse}. \tag{25}$$

5.3.1 Over-maintenance cost

Maintenance should be avoided when trains can still run for some time/distance because this generates additional production costs. Hence, for any maintenance operation, the remaining DBM (expressed in seconds) or TBM (expressed in km) of the concerned train is penalized. The corresponding costs, $remDCost$ and $remTCost$ are expressed per second and per km, respectively.

$$f_3^{maint} = \sum_{\substack{e \in \mathcal{E} \\ y_e = \texttt{BegMaintenance} \text{ and } c_e = \text{“D”}}} remDCost.remDBM_{t_e} + \sum_{\substack{e \in \mathcal{E} \\ y_e = \texttt{BegMaintenance} \text{ and } c_e = \text{“T”}}} remTCost.remTBM_{t_e} \quad (26)$$

5.3.2 Train junction / disjunction operation cost

Each junction and disjunction operation has a cost in the objective function, f_3^{jun} is the sum of these individual costs:

$$f_3^{jun} = \sum_{\substack{e \in \mathcal{E} \\ y_e = \texttt{BegJunction}}} junCost + \sum_{\substack{e \in \mathcal{E} \\ y_e = \texttt{BegDisjunction}}} disjCost. \quad (27)$$

5.3.3 Platform usage costs

A platform may be used in four cases:

1. For an `Arrival` event only (let $\mathcal{A}^* \subseteq \mathcal{A}$ be the set of such arrivals),
2. for a `Departure` event only (let $\mathcal{D}^* \subseteq \mathcal{D}$ be the set of such departures),
3. for an `Arrival` event immediately followed in $sched_t$ by a `Departure` event ($\mathcal{Z} \subseteq \mathcal{A} \times \mathcal{D}$ represents the set of such arrival/departure pairs where t does not leave the platform), or
4. for none of these cases.

Some costs apply to cases 1, 2 and 3; the associated cost functions are respectively denoted by $f_3^{plat_1}$, $f_3^{plat_2}$ and $f_3^{plat_3}$, where $dwellCost$ represents the cost of one second of variation between the ideal stay duration and the actual stay duration on a platform. Note for any pair $(a, d) \in \mathcal{Z}$ we have $dwell_a = dwell_d$.

For any arrival $a \in \mathcal{A}^*$, let $dwell_a$ be the duration of use of the platform assigned to a (i.e. the time difference between the `EnterResource` and `ExitResource` events before and after a). Similarly, for any departure $d \in \mathcal{D}^*$, let $dwell_d$ be the duration of use of the platform assigned to d. And let $dwell_z$ and $idealDwell_z$ respectively denote the duration of use of a platform by a pair

$z = (a, d) \in \mathcal{Z}$, and the ideal duration defined as the sum of ideal durations of a and d:

$$idealDwell_z = idealDwell_a + idealDwell_d. \tag{28}$$

Then, we define:

$$f_3^{plat_1} = \sum_{a \in \mathcal{A}^*} dwellCost \times |dwell_a - idealDwell_a|, \tag{29}$$

$$f_3^{plat_2} = \sum_{d \in \mathcal{D}^*} dwellCost \times |dwell_d - idealDwell_d|, \text{ and} \tag{30}$$

$$f_3^{plat_3} = \sum_{(a,d) \in \mathcal{Z}} dwellCost \times |dwell_z - idealDwell_z|. \tag{31}$$

Finally, we set:

$$f_3^{plat} = f_3^{plat_1} + f_3^{plat_2} + f_3^{plat_3}. \tag{32}$$

5.3.4 Non-satisfied preferred platform assignment cost

For any arrival $a \in \mathcal{A}$, if the platform $platf_a$ assigned to a is not in $prefPlat_a$, a cost of $platAsgCost$ applies. Likewise, this cost applies if, for any departure $d \in \mathcal{D}$, the platform $platf_d$ assigned to d does not belong to $prefPlat_d$.

$$f_3^{pref} = \sum_{\substack{a \in \mathcal{A} \\ platf_a \notin prefPlat_a}} platAsgCost + \sum_{\substack{d \in \mathcal{D} \\ platf_d \notin prefPlat_d}} platAsgCost. \tag{33}$$

5.3.5 Non-satisfied train reuse cost

Finally, if some reuse $u \in \mathcal{U}$ is not satisified, a cost of $reuseCost$ applies. f_3^{reuse} is defined by:

$$f_3^{reuse} = \sum_{\substack{u \in \mathcal{U} \\ depTrain_{dep_u} \neq arrTrain_{arr_u}}} reuseCost. \tag{34}$$

6 Appendices

6.1 Typical volume of data

Typical volumes of data in the instances are given below. They are mostly provided as an indication but the largest figures can be considered as upper bounds:

- Number of days in the horizon: 1 to 14.
- Initial trains: 10 to 100.
- Arrivals: 50 to 500 per day.

- Departures: 50 to 500 per day.
- Train categories: 1 to 30.
- Preferred train reuse: 0% to 100% of departures.
- Linked Departure-Arrival: 0% to 100% of arrivals.
- Joint-arrival and joint-departures: 0% to 100%.
- Single tracks: 0 to 50.
- Platforms: 10 to 50.
- Maintenance facilities: 5 to 50.
- Track groups: 5 to 20.
- Yards: 1 to 5.

6.2 Files format

6.2.1 Input files

Input files, containing all required data for one particular instance, are provided in CSV format (comma-separated values), with ";" as separator and the first line containing headers (name of columns). These files have the following names:

- arrivals.csv
- departures.csv
- initialTrains.csv
- trainCategories.csv
- arrDepSequences.csv
- prefPlat.csv
- compCatDep.csv
- compCatRes.csv
- reuses.csv
- jointArrDep.csv
- gates.csv
- singleTracks.csv
- platforms.csv

- facilities.csv
- trackGroups.csv
- yards.csv
- imposedConsumptions.csv
- parameters.csv

For the sake of convenience, another file called inputData.csv will be provided. It will be a concatenation of all input files.

6.2.2 Output files

Solutions should be provided as one single CSV file for each instance. The first line of each solution file is supposed to be dedicated to headers, therefore it will not be read and should not contain data. The lines in the solution file each correspond to one event and must be sorted by trains first, and then by transition times. When two events on the same train occur at the same time (for instance, `Arrival` events occur at the same time as `EnterResource` on platforms), `EnterSystem` must come first, then `EnterResource`, then any other type, and `ExitSystem` come last, after `ExitResource`, itself after any other type. An example of schedules is presented in Tables 4 to 6. In this exemple, Train1 performs a junction operation with Train9 and Train12 (already assembled), in preparation of a joint-departure composed of Departure34, Departure35 and Departure36. Once assembled, the three trains perform a maintenance operation of type "D"on Facility1 before reaching Platform14, and leave the system through TrGroup9.

Any event e is associated with a train, t_e, a time, h_e, an event type y_e, a resource, r_e, a gate on r_e, g_e, and a complement, c_e. All attributes do not need to be provided in all cases. In the following, when attributes of e are not defined, it is assumed that the corresponding fields should remain blank in the solution file.

The type y_e of event e can be any of:

- `EnterSystem`: entrance of the train in the system. This event is compulsory for each train, it must be the first one of each schedule.

  If t is initially in the system:

  - $h_e = d_1$ 00 : 00 : 00.
  - r_e is the resource used by train t at the beginning of the horizon: $r_e = res_t$.
  - g_e is not defined.
  - c_e is not defined.

  If t is associated with an arrival a:

Train	Time	Event type	Resource	Gate	Complement
Train1	d_2 06:45:00	EnterSystem	TrGroup5		
Train1	d_2 06:45:00	EnterResource	TrGroup5	A3	
Train1	d_2 06:49:00	ExitResource	TrGroup5	B1	
Train1	d_2 06:49:00	EnterResource	Platform12	A1	
Train1	d_2 06:49:00	Arrival	Platform12		Arrival26
Train1	d_2 07:18:00	ExitResource	Platform12	A1	
Train1	d_2 07:18:00	EnterResource	TrGroup6	B4	
Train1	d_2 07:23:00	ExitResource	TrGroup6	A3	
Train1	d_2 07:23:00	EnterResource	Yard5	A2	
Train1	d_2 07:35:00	BegJunction	Yard5		Train1
Train1	d_2 07:38:00	EndJunction	Yard5		Train1+Train9+Train12
Train1	d_2 09:02:00	ExitResource	Yard5	B3	
Train1	d_2 09:02:00	EnterResource	TrGroup7	B1	
Train1	d_2 09:04:00	ExitResource	TrGroup7	A2	
Train1	d_2 09:04:00	EnterResource	TrGroup8	A1	
Train1	d_2 09:09:00	ExitResource	TrGroup8	B2	
Train1	d_2 09:09:00	EnterResource	Facility1	A1	
Train1	d_2 09:09:00	BegMaintenance	Facility1		"D"
Train1	d_2 11:09:00	EndMaintenance	Facility1		"D"
Train1	d_2 11:45:00	ExitResource	Facility1	A1	
Train1	d_2 11:45:00	EnterResource	TrGroup8	B3	
Train1	d_2 11:50:00	ExitResource	TrGroup8	A8	
Train1	d_2 11:50:00	EnterResource	Platform14	A1	
Train1	d_2 12:20:00	Departure	Platform14		Departure34
Train1	d_2 12:20:00	ExitResource	Platform14	A1	
Train1	d_2 12:20:00	EnterResource	TrGroup9	A2	
Train1	d_2 12:25:00	ExitResource	TrGroup9	B2	
Train1	d_2 12:25:00	ExitSystem	TrGroup9		

Table 4: Example: schedule of Train1

Train	Time	Event type	Resource	Gate	Complement
...	...	...	...	...	...
Train9	d_2 07:35:00	BegJunction	Yard5		Train9+Train12
Train9	d_2 07:38:00	EndJunction	Yard5		Train1+Train9+Train12
Train9	d_2 09:02:00	ExitResource	Yard5	B3	
Train9	d_2 09:02:00	EnterResource	TrGroup7	B1	
Train9	d_2 09:04:00	ExitResource	TrGroup7	A2	
Train9	d_2 09:04:00	EnterResource	TrGroup8	A1	
Train9	d_2 09:09:00	ExitResource	TrGroup8	B2	
Train9	d_2 09:09:00	EnterResource	Facility1	A1	
Train9	d_2 09:09:00	BegMaintenance	Facility1		"D"
Train9	d_2 11:09:00	EndMaintenance	Facility1		"D"
Train9	d_2 11:45:00	ExitResource	Facility1	A1	
Train9	d_2 11:45:00	EnterResource	TrGroup8	B3	
Train9	d_2 11:50:00	ExitResource	TrGroup8	A8	
Train9	d_2 11:50:00	EnterResource	Platform14	A1	
Train9	d_2 12:20:00	Departure	Platform14		Departure35
Train9	d_2 12:20:00	ExitResource	Platform14	A1	
Train9	d_2 12:20:00	EnterResource	TrGroup9	A2	
Train9	d_2 12:25:00	ExitResource	TrGroup9	B2	
Train9	d_2 12:25:00	ExitSystem	TrGroup9		

Table 5: Example: schedule of Train9 (extract)

Train	Time	Event type	Resource	Gate	Complement
...	...	...	...	...	...
Train12	d_2 07:35:00	BegJunction	Yard5		Train9+Train12
Train12	d_2 07:38:00	EndJunction	Yard5		Train1+Train9+Train12
Train12	d_2 09:02:00	ExitResource	Yard5	B3	
Train12	d_2 09:02:00	EnterResource	TrGroup7	B1	
Train12	d_2 09:04:00	ExitResource	TrGroup7	A2	
Train12	d_2 09:04:00	EnterResource	TrGroup8	A1	
Train12	d_2 09:09:00	ExitResource	TrGroup8	B2	
Train12	d_2 09:09:00	EnterResource	Facility1	A1	
Train12	d_2 09:09:00	BegMaintenance	Facility1		"D"
Train12	d_2 11:09:00	EndMaintenance	Facility1		"D"
Train12	d_2 11:45:00	ExitResource	Facility1	A1	
Train12	d_2 11:45:00	EnterResource	TrGroup8	B3	
Train12	d_2 11:50:00	ExitResource	TrGroup8	A8	
Train12	d_2 11:50:00	EnterResource	Platform14	A1	
Train12	d_2 12:20:00	Departure	Platform14		Departure36
Train12	d_2 12:20:00	ExitResource	Platform14	A1	
Train12	d_2 12:20:00	EnterResource	TrGroup9	A2	
Train12	d_2 12:25:00	ExitResource	TrGroup9	B2	
Train12	d_2 12:25:00	ExitSystem	TrGroup9		

Table 6: Example: schedule of Train12 (extract)

- h_e must be the arrival time minus the travel time required to perform the arrival sequence, throughout the successive track groups:

$$h_e = arrTime_a - \sum_{k \in arrSeq_a} trTime_k. \tag{35}$$

- r_e must be first track group of the arrival sequence of a.
- g_e is not defined.
- c_e is not defined.

- `ExitSystem`: exit of the train from the system. This event is also compulsory for each train, it must be the last one of the schedule.

  If t is assigned to a departure d:

  - h_e must be the time of exit of the last track group of the departure sequence of d:

$$h_e = depTime_d + \sum_{k \in depSeq_d} trTime_k. \tag{36}$$

  - r_e must be the last track group of $depSeq_d$.
  - g_e is not defined.
  - c_e is not defined.

  If it is not assigned to any departure:

- $h_e = d_{nbDays}$ 23 : 59 : 59.
- r_e is the resource used by t at the end of the horizon.
- g_e is not defined.
- c_e is not defined.

- `Arrival`: arrival a associated with t_e, if t_e is not initially in the system.
  - h_e is the arrival time: $h_e = arrTime_a$.
  - r_e is the platform p assigned to a.
  - g_e is not defined.
  - c_e is the identifier of the arrival a.

- `Departure`: departure d to which train t is assigned, if any.
  - h_e is the departure time: $h_e = depTime_d$.
  - r_e is the platform p assigned to d.
  - g_e is not defined.
  - c_e is the identifier of the departure d.

- `EnterResource`: entrance of a train on a resource of the system.
  - h_e is the time of entrance. It is the beginning of use of r_e by t.
  - r_e is resource used by t from h_e.
  - g_e is the gate through which t arrives on r_e.
  - c_e is not defined.

- `ExitResource`: exit of a train from a resource.
  - h_e is time of exit. It is the end of use of r_e by t.
  - r_e is the resource used by t until h_e.
  - g_e is the gate through which t leaves r_e. It is recalled that every gate has exactly one neighbor gate: hence, g_e must be the neighbor of the gate g' through which t enters on r', next resource used by t (except if t leaves the system through g_e, in which case ge has no neighbor).
  - c_e is not defined.

- `BegJunction` and `EndJunction`: respectively, beginning and end of a junction operation with one or more other trains. An `EndJunction` event must always follow immediately a `BegJunction` event, with a difference in their times corresponding to the duration of the junction operation.
  - h_e is the time of beginning (if y_e is `BegJunction`) or end (if y_e is `EndJunction`) of the junction operation.

  - r_e is the resource on which the junction is performed. Recall that a junction operation can only be performed on platforms, single tracks, maintenance facilities or yards (see Section 2.5).
  - g_e is not defined.
  - c_e is the list of assembled trains, sorted by position with respect to side A of r_e (the train located most on side A of r_e comes first). For `BegJunction`, this list is made up of t and trains assembled with t, if any, *before* the junction operation; for `EndJunction`, it is made up of all trains assembled *after* the junction operation (hence, c_e must strictly include all trains in $c_{e'}$ of the previous `BegJunction` event e'). The identifiers of trains should be separated by "+" characters.

- `BegDisjunction` and `EndDisjunction`: respectively, beginning and end of a disjunction operation with one or more other trains. An `EndDisjunction` event must always follow immediately a `BegDisjunction` event, with a difference in their times corresponding to the duration of the disjunction operation.
  - h_e is the time of beginning (if y_e is `BegDisjunction`) or end (if y_e is `EndDisjunction`) of the disjunction operation.
  - r_e is the resource on which the disjunction is performed. Disjunction operations can only be performed on platforms, single tracks, maintenance facilities or yards.
  - g_e is not defined.
  - c_e is the list of assembled trains, sorted by position with respect to side A of r_e (the train located most on side A of r_e comes first). For `BegDisjunction`, this list is made up of t and trains assembled with t, if any, *before* the disjunction operation; for `EndDisjunction`, it is made up of all trains still assembled after the disjunction operation (hence, $c_{e'}$ of the previous `BegDisjunction` event e' must strictly include all trains in c_e). The identifiers of trains should be separated by "+" characters.

- `BegMaintenance` and `EndMaintenance`: respectively, beginning and end of a maintenance operation on the train. An `EndMaintenance` event must always follow immediately a `BegMaintenance` event, with a difference in their times corresponding to the duration of the maintenance operation.
  - h_e is the time of beginning (if y_e is `BegMaintenance`) or end (if y_e is `EndMaintenance`) of the maintenance operation.
  - r_e is the maintenance facility on which the maintenance operation is performed.
  - g_e is not defined.
  - c_e is "D" for a maintenance operation of type "D" or "T" for an operation of type "T". Obviously, the type of potential reset must be coherent with the associated maintenance facility, r_e.

List of symbols

$\mathcal{A}$	Set of arrivals
$\mathcal{C}$	Set of train categories
$\mathcal{D}$	Set of departures
$\mathcal{E}$	Set of events in the solution
$\mathcal{F}$	Set of maintenance facilities
$\mathcal{G}_r$	Set of gates of resource r
$\mathcal{H}$	Set of time instants in the planning horizon
$\mathcal{I}$	Set of pre-defined resource consumptions
$\mathcal{J}_{arr}$	Set of joint-arrivals
$\mathcal{J}_{dep}$	Set of joint-departures
$\mathcal{K}$	Set of track groups
$\mathcal{P}$	Set of platforms
$\mathcal{R}$	Set of resources
$\mathcal{S}$	Set of single tracks
$\mathcal{T}$	Set of trains
$\mathcal{T}_A$	Set of trains associated with arrivals
$\mathcal{T}_I$	Set of trains initially in the system
$\mathcal{U}$	Set of preferred reuses
$\mathcal{Y}$	Set of yards
$arrSeq_a$	Arrival sequence of arrival a
$arrTime_a$	Arrival time of arrival a
$arrTrain_a$	Train associated with arrival a
arr_u	Arrival of reuse u
beg_i	Beginning time of imposed consumption i
$capa_s$	Capacity of single track s
$capa_y$	Capacity of yard y
$catGroup_c$	Compatibility group of train category c
cat_t	Category of train t
c_e	(*) Complement of event e
$compCatDep_d$	Set of train categories compatible with departure d
$compCatRes_r$	Set of train categories compatible with resource r
$depSeq_d$	Departure sequence of departure d
$depTime_d$	Departure time of departure d
$depTrain_d$	(*) Train assigned to departure d, if any
dep_u	Departure of reuse u
$disjCost$	Cost associated with a disjunction operation
$disjTime$	Time required to perform a disjunction operation
$dwellCost$	Cost associated with every second of difference between ideal and actual time of stay on a platform, for an arrival or a departure
end_i	End time of imposed consumption i
g_e	(*) Gate of event e
$hwTime_k$	Headway time on track group k
h_e	(*) Time of event e

$idealDwell_a$	Ideal dwell time on platform after arrival a
$idealDwell_d$	Ideal dwell time on platform before departure d
ind_g	Index of gate g
$jaList_j$	Ordered list of arrivals for joint-arrival j
$jdList_j$	Ordered list of departures for joint-departure j
$jointArr_a$	Joint-arrival arrival a belongs to, if any
$jointDep_d$	Joint-departure departure d belongs to, if any
$junCost$	Cost associated with a junction operation
$junTime$	Time required to perform a junction operation
$length_r$	Length of resource r
$length_t$	Length of train t
$linkedDep_a$	Departure linked with arrival a
$maintD_{t,i}$	(*) Equals to 1 if train t performs a maintenance operation of type "D"on day i, 0 otherwise
$maintTimeD_c$	Time required to perform a maintenance operation of type "D"on trains of train category c
$maintTimeT_c$	Time required to perform a maintenance operation of type "T"on trains of train category c
$maintT_{t,i}$	(*) Equals to 1 if train t performs a maintenance operation of type "T"on day i, 0 otherwise
$maxDBM_c$	Maximum DBM of train category c
$maxDwellTime$	Maximum duration of use of a platform in absence of an arrival or a departure
$maxDwell_a$	Maximum dwell time on platform after arrival a
$maxDwell_d$	Maximum dwell time on platform before departure d
$maxMaint$	Maximum number of maintenance operations per day
$maxTBM_c$	Maximum TBM of train category c
$minAsbTime$	Minimum assembled time: minimum duration before departure during which trains must assembled for a joint-departure
$minResTime$	Minimum duration of use of a resource
$reuseCost$	Cost associated with a preferred reuse not satisified in the solution
$nbDays$	Number of days in the horizon
$nbTrain_{r,h}$	(*) Number of trains using resource r a time h
nb_i	Number of trains (on a yard) of imposed consumption i
$neighSet_r$	Set of neighbor resources of resource r
$neigh_g$	Neighbor gate of gate g
$platAsgCost$	Cost associated with a non-satisfied preferred platform assignment
$prefPlat_a$	Preferred platform for arrival a
$prefPlat_d$	Preferred platform for departure d
$remDBM_a$	Remaining DBM of train associated with a
$remDBM_t$	Remaining DBM of train t
$remDCost$	Cost associated with every km of remaining DBM when a maintenance operation is started on a train
$remTBM_a$	Remaining TBM of train associated with a

$remTBM_t$	Remaining TBM of train t
$remTCost$	Cost associated with every second of remaining TBM when a maintenance operation is started on a train
$reqDBM_d$	Required DBM for departure d
$reqTBM_d$	Required TBM for departure d
res_i	Resource of imposed consumption i
res_t	Resource used at h_0 by train t initially in the system
$revTime$	Reverse time: minimum duration a train must remain on a resource if its entrance side and its exit side on this resource are opposite
r_e	(*) Resource of event e
r_g	(*) Resource of gate g
$sched_t$	(*) Schedule of train t
$side_g$	Side of gate g
t_e	(*) Train of event e
$trTime_k$	Travel time on track group k
$type_f$	Type of maintenance performed by facility f
$use_{t,h}$	(*) Resource used by train t at h
y_e	(*) Type of event e

Note: (*) means that the notation is used for a decision variable, as opposed to the other notations wich are parameters of the problem.