Rescheduling Railway Traffic on Real Time Situations using Time-Interval Variables

Quentin Cappart and Pierre Schaus

Université catholique de Louvain, Louvain-La-Neuve, Belgium
{quentin.cappart|pierre.schaus}@uclouvain.be

Abstract. In the railway domain, the action of directing the traffic in accordance with an established timetable is managed by a software. However, in case of real time perturbations, the initial schedule may become infeasible or suboptimal. Subsequent decisions must then be taken manually by an operator in a very limited time in order to reschedule the traffic and reduce the consequence of the disturbances. They can for instance modify the departure time of a train or redirect it to another route. Unfortunately, this kind of hazardous decisions can have an unpredicted negative snowball effect on the delay of subsequent trains. In this paper, we propose a Constraint Programming model to help the operators to take more informed decisions in real time. We show that the recently introduced time-interval variables are instrumental to model this scheduling problem elegantly. We carried experiments on a large Belgian station with scenarios of different levels of complexity. Our results show that the CP model outperforms the decisions taken by current greedy strategies of operators.

1 Introduction

Since the dawn of the nineteenth century, development of railway systems has taken a huge importance in many countries. Over the years, the number of trains, the number of tracks, the complexity of networks increase and are still increasing. In this context, the need of an efficient and reliable train schedule is crucial. Indeed, a bad schedule can cause train conflicts, unnecessary delays, financial losses, and a passenger satisfaction decrease. While earliest train schedules could be built manually without using optimisation or computer based methods, it is not possible anymore. Plenty of works deal with this problematic of building the most appropriate schedule for general [1–3], or specific purposes [4, 5].

Practical schedules must also deal with real time perturbations. Disturbances, technical failures or simply consequences of a too optimistic theoretical schedule can cause delays which can be propagated on other trains. The initial schedule may then become infeasible. Real-time modifications of the initial schedule therefore may be required. Several works already tackle this problem. A recent survey (2014) initiated by Cacchiani et al. [6] recaps the different trends on models and algorithms for real-time railway disturbance management. For instance, Fay et al. [7] propose an expert system using fuzzy rules and Petri Net for the modelling. Such a method requires to define the rules, which can differ according to the station. Higgins et al. [8] propose to use local search methods in order to solve conflicts. However, this work does not take into account the alternative routes that trains can have in order to reach their destination, and that the planned route remains not always optimal in case of real time perturbations. For that, D'Ariano et al. [9] model the problem with an alternative graph formulation and propose a branch and bound algorithm to solve it. They suggest to enrich their method with a local search algorithm for rerouting optimization purposes [10]. Besides, some works consider the passenger satisfaction [11–13] in their model. It is also known as the Delay Management Problem. Generally speaking, most of the methods dealing with train scheduling are based on Mixed Integer Programming [14–17]. However, the performance of Mixed Integer Programming models for solving scheduling problems is known to be highly dependant of the granularity of time chosen.

Constraint Based Scheduling [18], or in other words, applying Constraint Programming on scheduling problems seems to be a good alternative over Mixed Integer Programming. According to the survey of Bartak et al. [19], Constraint Programming is particularly well suited for real-life scheduling applications. Furthermore, several works [20–22] show that Constraint Programming can be used for solving scheduling problems on large and realistic instances. By following this trend, Rodriguez [23] proposes a Constraint Programming model for real-time train scheduling at junctions. However, despite the good performances obtained, this model can be improved. Firstly, the modelling do no use the strength of global constraints which can provide a better propagation. Secondly, the search can also be improved with heuristics and the use of Local Search techniques. Finally, the objective function is only defined in function of train delays without considering the passengers or the different categories of trains.

In this paper, we propose a new model for rescheduling the railway traffic in a real time context. The contributions are as follows:

- A Constraint Programming model for rescheduling the traffic through real time disturbances on the railway network. In addition to being a relevant application for railway companies, the proposed model scales on a realistic instance, the station of Courtrai (Belgium).
- The application of state of the art scheduling algorithms and relevant global constraints in order to achieve a better propagation and a faster search. Concretely, the conditional time-intervals introduced by Laborie et al. [24, 25] as well as their dedicated global constraints have been used. Furthermore, the exploration of the state space has been carried out using Failure Directed Search together with Large Neighbourhood Search [26].
- The formulation of an objective function aiming at the same time to minimise the total delay and to maximise the overall passenger satisfaction. Furthermore, the objective function also considers the heterogeneity of the

traffic and different levels of priority between trains through a defined lexicographical order. For instance a freight train has a lower priority than a passenger train.

The implementation of the model has been performed with IBM ILOG CP Optimizer V12.6.3 [27] which is particularly fitted for designing scheduling models [28–30]. The next section describes the signalling principles and the components considered for the modelling. Section 3 introduces the model and its specificities. Experiments and discussions about its performances are then carried in Section 4.

2 Signalling principles

In the railway domain, the goal of a signalling system is to ensure a safe control of the traffic [31]. Such a task encompasses the regulation of the traffic. In Belgium, it is mainly performed by a software, called the Traffic Management System, that automatically regulates the traffic according to predefined rules. Let us consider the fictive station presented in Fig. 1. Several components are depicted on it:

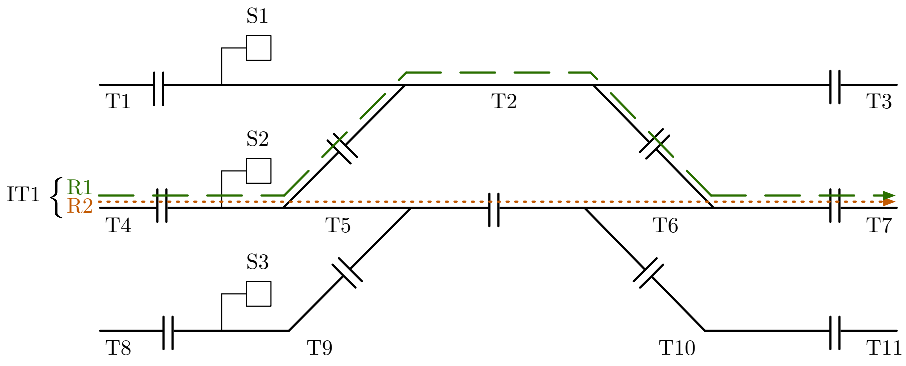


Fig. 1: Track layout of a fictive station with two routes ($R1$ and $R2$).

- The **track segments** (e.g. $T1$) are the portions of the railway structure where a train can be detected. They are delimited by the **joints** ($\dashv\vdash$).
- The **signals** (e.g. $S1$) are the devices used to control the train traffic. They are set on a proceed state (green) if a train can safely move into the station or in a stop state (red) otherwise.

Besides these physical components, signalling also involves logical structures:

- The **routes** correspond to the paths that trains can follow inside a station in order to reach a destination. They are expressed in terms of track segments and signals. For instance, $R1$ is a route going from $T4$ to $T7$ by following the path $[T4, S2, T5, T2, T6]$. The first track segment of a route is always in front of a signal which is initially at a stop state and which turns green when the Traffic Management System allows the train to proceed. The track segment used for the destination is not a part of the route. This action is called a **route activation**. At this step, all the track segments forming the route after the start signal are reserved and cannot be used by another train. Once a route has been activated, the train can move through it in order to reach its destination. For instance, once a train following route $R1$ has reached $T6$ and is not on the previous track segments anymore, $T5$ and $T2$ can be released in order to allow other trains to use them. Detailed explanations about the route management is provided in [32].
- The **itineraries** correspond to non physical paths from a departure point to an arrival point. An itinerary can be constituted of one or several routes which can be alternative. For instance, as depicted in Fig. 1, two routes ($R1$ and $R2$) are possible in order to accomplish itinerary $IT1$ from $T4$ to $T7$. In normal situations, a route is often preferred than others, but in case of perturbations, the route causing the less conflicts is preferred.

According to an established timetable, the Traffic Management System activates routes in order to ensure the planned traffic. However, in case of real time perturbations, signalling must be handled manually and the actions to perform are decided by human operators. The procedure follows this pattern:

1. The Traffic Management System has predicted a future conflict caused by perturbations on the traffic.
2. The operators controlling the traffic analyse the situation and evaluate the possible actions to do in order to minimise the consequences of the perturbations. Besides the safety, the criterion considered for the decision is the sum of delays per passenger. In other words, the objective is to minimise the sum of delays of trains pondered by their number of passenger. Furthermore, some trains can have a higher priority than others.
3. According to the situation, they perform some actions (stopping a train, changing its route, etc.) or do nothing.

In this work, we are interested by the actions of operators facing up real time perturbations. In such situations, they must deal with several difficulties:

- The available time for analysing the situation and taking a decision can be very short according to the criticality of the situation (less than one minute).
- For large stations with a dense traffic, particularly during the peak hours, the effects of an action are often difficult to predict.
- The number of parameters that must be considered (type of trains, number of passengers, etc.) complicates the decision.

Such reasons can lead railway operators to take decisions that will not improve the situation, or worse, will degrade it. Most of the time, the decision taken is to give the priority either to the first train arriving at a signal, or to the first one that must leave the station [9]. However, it is not always the best decision. It is why we advocate the use of a Decision Support Tool based on optimisation in order to assist operators in their decisions. The requirement for this software is then to provide a good solution to this rescheduling problem within a short and parametrisable computation time.

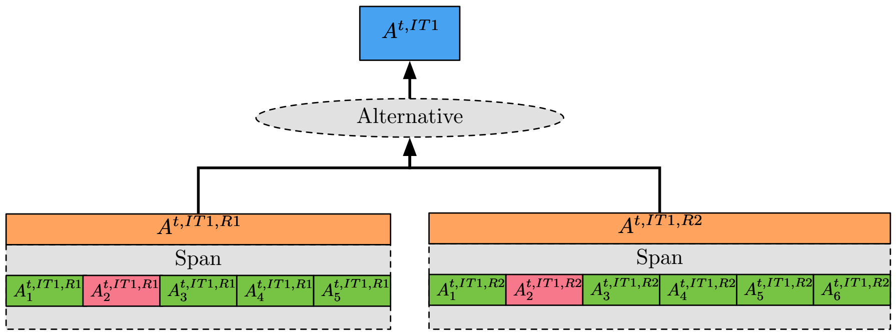
3 Modelling

This section presents how we model the problem. Basically, the goal is to schedule adequately trains in order to bring them to their destination. The decision is then to chose, for each train, which route must be activated and at what time. Each track segment can host at most one train at a time. Furthermore, they can also be reserved only for one train. An inherent component of scheduling problems are the activities. Roughly speaking, a classical activity A is modelled with three variables, a start date $s(A)$, a duration $d(A)$ and an end date $e(A)$. The activity is *fixed* if the three variables are reduced to a singleton. Our model contains three kinds of activities that are linked together:

- The **itinerary activities** define the time interval when a train follows a particular itinerary. Each train has one and only one itinerary activity. We define $A^{t,it}$ as the activity for itinerary it of train t.
- The **route activities** define the interval when a train follows a particular route of an itinerary. We define $A^{t,it,r}$ as the activity for route r of itinerary it related to train t.
- The **train activities** correspond to the movements of a train through the station in order to complete a route. Such activities use the track segments as resources. We define $A_i^{t,it,r}$ as the i^{th} train activity of route r of itinerary it and related to train t. Inside a same route, there are as many train activities as the number of elements on the route path. The element can be a track segment or a signal. For instance, by following the example of Fig. 1, $A_1^{t,IT1,R1}$ is a train activity related to track segment $T4$, $A_2^{t,IT1,R1}$ to signal $S2$, $A_3^{t,IT1,R1}$ to $T5$, etc.

One particularity of our problem is that some activities are optional. In other words, they may or may not be executed in the final schedule. For instance, let us assume that a train has to accomplish an itinerary from $T4$ to $T7$. To do so, it can follow either $R1$, or $R2$. If $R1$ is chosen, the activity related to $R2$ will not be executed. For that, we model optional activities with the *conditional time-interval variables* introduced by Laborie et al. [24, 25] that implicitly encapsulate the notion of optionality. It also allows an efficient propagation through dedicated global constraints (`alternative` and `span` for instance) as well as an efficient search. Roughly speaking, when an activity is fixed, it can be either *executed*, or *non executed*. If the activity is executed, it behaves as a classical activity that executes on its time interval, otherwise, it is not considered by any constraint. This functionality is modelled with a new variable $x(A) \in \{0, 1\}$ for each activity A such that $x(A) = 1$ if the activity is executed and $x(A) = 0$ otherwise.

Fig. 2 presents how the different activities are organised for the case study shown in Fig. 1 for a train t. Activity $A^{t,IT1}$ is mandatory, it models the fact that the train has to reach its destination. `Alternative` constraint ensures that the train can follow only one route to do so, $R1$ or $R2$. The other route activity is not executed. Furthermore, the start date and the end date of the chosen route is synchronised with the itinerary activity. `Span` constraint enforces the route activity to be synchronised with the start date of the first track segment activity and with the end date of the last track segment activity. For instance, $A^{t,IT1,R1}$ is synchronised with the start date of $A_1^{t,IT1,R1}$ and the end date of $A_5^{t,IT1,R1}$. The detailed explanations of `alternative` and `span` constraints are provided thereafter.



$A^{t,IT1}$
Alternative
$A^{t,IT1,R1}$
Span
$A_1^{t,IT1,R1}$ $A_2^{t,IT1,R1}$ $A_3^{t,IT1,R1}$ $A_4^{t,IT1,R1}$ $A_5^{t,IT1,R1}$
$A^{t,IT1,R2}$
Span
$A_1^{t,IT1,R2}$ $A_2^{t,IT1,R2}$ $A_3^{t,IT1,R2}$ $A_4^{t,IT1,R2}$ $A_5^{t,IT1,R2}$ $A_6^{t,IT1,R2}$

Fig. 2: Breakdown structure of the model using `alternative` and `span` constraints.

As we will see, conditional time-interval variables facilitate the construction of our model. Let us now define the different components of the model.

Parameters Two entities are involved in our model: the trains and the track segments. Table 1 recaps the parameters considered. The *speed*, *number of passengers*, and *length* are straightforward to understand. The *estimated arrival time* of a train is a prediction of its arrival time at the station. The *earliest start time* defines a lower bound on the starting time of a train. In other words, a train cannot start its itinerary before this time. Indeed, a train cannot leave its platform before the time announced to the passengers. The *planned completion time* is the time announced on the initial schedule. It defines when the train is supposed to arrive at a platform. It is used in the objective function in order to compute the delays generated. The *category* defines the nature of the train. More explanations about the category is provided in Section 3.

Table 1: Parameters related to a train t or a track segment ts.

Entity	**Parameter**	**Name**	**Meaning**
Train	Speed	spt_t	Speed of t
	Passengers	p_t	Number of passengers of t
	Estimated Arrival Time	eat_t	When t arrives to the station
	Earliest Start Time	est_t	Lower bound on start time of t
	Planned Completion Time	pct_t	When t must complete its journey
	Category	cat_t	Category of t
Track Segment	Length	lgt_{ts}	Length of ts

Decision Variables As previously said, the problem is to chose, for each train, which route must be activated and at what time. Such a problem can be seen as a slightly variant of a job shop scheduling problem [33] where the machines represent the track segments and the jobs represent train activities. The difference are as follows:

- Some activities are optional. In other words, they may or not be executed in the final schedule.
- The end of a train activity must be synchronised by the start of the next one.
- A train activity can use more than one resource. When a train is on a particular track segment, its current activity uses the current track segment as well as the next ones that are reserved for the route. For instance, let us consider the route activity $A^{t,IT1,R1}$. The related train activities with their resources are $A_1^{t,IT1,R1}, A_2^{t,IT1,R1}, A_3^{t,IT1,R1}, A_4^{t,IT1,R1}$ and $A_5^{t,IT1,R1}$. The resources used by the activities are $(T4), (T4, S2), (T5, T2, T6), (T2, T6)$ and $(T6)$ respectively. Let us remember from Section 2 that only the track segments located after the start signal are reserved through the route activation. Concerning the initial track segment T_4, it is released after the train has passed the start signal.

From a Constraint Based Scheduling approach, the problem is to assign a unique value to each train activity. The decision variables and their domain are, for all trains t, itineraries it, routes r and indexes i:

$$s(A_i^{t,it,r}) \begin{cases} \in [eat_t, horizon] & if\ t\ on\ track\ segment\ ts \\ \in [est_t, horizon] & if\ t\ in\ front\ of\ a\ signal \end{cases} \quad (1)$$

$$d(A_i^{t,it,r}) \begin{cases} = lgt_{ts}/spd_t & if\ t\ on\ track\ segment\ ts \\ \in [0, horizon] & if\ t\ in\ front\ of\ a\ signal \end{cases} \quad (2)$$

$$e(A_i^{t,it,r}) = s(A_i^{t,it,r}) + d(A_i^{t,it,r}) \tag{3}$$

$$x(A_i^{t,it,r}) \in \{0,1\} \tag{4}$$

The domain is determined in order to be as restricted as possible without removing a solution. Equation (1) indicates that an activity cannot begin before the *estimated arrival time* of t. The upper bound of the start date is defined by the time horizon considered. More details about the horizon chosen is provided in Section 4. Equation (2) models the time required to achieve the activity .If t is on a track segment, the duration is simply the length of the track segment divided the speed of t. Otherwise, the time that t will have to wait is unknown. Equation (3) is an implicit constraint of consistency. Finally, Equation (4) states that the activity is optional. Concerning route and itinerary activities, they are linked to train activities through constraints.

Constraints This section describes the different constraints considered. Most of them are expressed in term of a train, an itinerary and a route. Let us express T as the set of trains, IT_t as the set of possible itineraries for $t \in T$, R_{it} the set of possible routes for $it \in IT_t$ and N_r as the number of train activities of a route $r \in R_{it}$. Furthermore, let us state TS as the set of all the track segments in the station.

Precedence This constraint (Equation (5)) ensures that train activities must be executed in a particular order. It links the end of a train activity $A_i^{t,it,r}$ with the start of $A_{i+1}^{t,it,r}$.

$$e(A_i^{t,it,r}) = s(A_{i+1}^{t,it,r}) \qquad \forall t \in T, \forall it \in IT_t, \forall r \in R_{it}, \forall i \in [1, N_r) \tag{5}$$

All precedence constraints are aggregated into a temporal network in order to have a better propagation [24, 34]. Instead of having a bunch of independent constraints, they are considered as a global constraint.

Execution Consistency As previously said, some activities are alternative. If a route is not chosen for a train, none of activities $A_i^{t,it,r}$ will be executed. Otherwise, all of them must be executed. Equation (6) states that all the train activities related to the same environment must have the same execution status.

$$x(A_1^{t,it,r}) \equiv x(A_i^{t,it,r}) \qquad \forall t \in T, \forall it \in IT_t, \forall r \in R_{it}, \forall i \in]1, N_r] \tag{6}$$

Alternative Introduced by Laborie and Rogerie [24], this constraint models an exclusive alternative between a bunch of activities. It is expressed in Equation (7).

$$\texttt{alternative}\Big(A^{t,it}, \Big\{A^{t,it,r}\Big| r \in R_{it}\Big\}\Big) \qquad \forall t \in T, \forall it \in IT_t \quad (7)$$

It means that when $A^{t,it}$ is executed, then exactly one of the route activities must be executed. Furthermore, the start date and the end date of $A^{t,it}$ must be synchronised with the start and end date of the executed route activity. if $A^{t,it}$ is not executed, none of the other activities can be executed. In our model, $A^{t,it}$ is a mandatory activity. It models the fact that each train must reach its destination through an itinerary but for that, it must follow exactly one route.

Span Also introduced in [24], this constraint states that an executed activity must span over a bunch of other executed activities by synchronising its start date with the earliest start date of other executed activities and its end date with the latest end date. It is expressed in Equation 8.

$$\texttt{span}\Big(A^{t,it,r}, \Big\{A_i^{t,it,r}\Big| i \in [1, N_r]\Big\}\Big) \qquad \forall t \in T, \forall it \in IT_t, \forall r \in R_{it} \quad (8)$$

It models the fact that the time taken by a train to complete a route is equal to the time required for crossing each of its components. If the route is not chosen, then no activity will be executed. A representation of `span` constraint is shown in Fig. 2.

Unary Resource An important constraint is that trains cannot move or reserve a track segment that is already used for another train. It is a `unary resource` constraint (Equation (9)). Each track segment can then be reserved only once at a time. Let us state ACT_{ts} as the set of all the train activities using track segment ts.

$$\texttt{noOverlap}\Big(\Big\{A\Big| A \in ACT_{ts}\Big\}\Big) \qquad \forall ts \in TS \quad (9)$$

The semantic of this constraint, as well as comparisons with existing frameworks, is presented in [25] which extends state of the art filtering methods in order to handle conditional time-interval variables.

Train Order Consistency This last constraint ensures that trains cannot overtake other trains if they are on the same track segment. An illustration of this scenario is presented in Fig. 3. Even if train t_2 has a higher priority than t_1, it cannot begin its activities before t_1 because t_1 has an earlier estimated arrival time. In other words, on each track segment in front of a signal, the start date of the first activity of each train is sorted by their estimated arrival time.

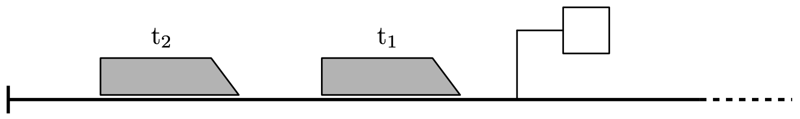


Fig. 3: Two trains waiting on the same track segment.

Let us state $TSB \subset TS$ as the set of all first track segments, N_{tsb} as the number of trains beginning their itinerary on track segment tsb, and (AB_i^{tsb}) as the sequence of the first activities of trains t_i beginning on tsb with $i \in [1, N_{tsb}]$. The sequence is ordered by the estimated arrival time of trains t_i. Then, `train order consistency` constraint can be expressed through Equation (10).

$$s(AB_{i-1}^{tsb}) < s(AB_i^{tsb}) \qquad \forall tsb \in TSB, \forall i \in]1, N_{tsb}] \tag{10}$$

These constraints are also considered in the temporal network.

Objective Function The criterion frequently used for the objective function is the sum of train delays [23]. Let us state jct_t as the *journey completion time* of a train t. It corresponds to the end date of the last train activity of t. The delay d_t of t is expressed in Equation 11.

$$d_t = \texttt{max}\Big(0, jct_t - pct_t\Big) \tag{11}$$

The `max` function is used to nullify the situation where t is in advance on its schedule. Equation 12 presents a first objective function.

$$\texttt{min}\Big(\sum_{t \in T} d_t\Big) \tag{12}$$

However, in real circumstances, railway operators must also consider other parameters such as the number of passengers and the priority of trains. Section 3 introduced the *category* parameter, here are the different categories considered:

1. Maintenance or special vehicles (C_1).
2. Passenger trains with a correspondence (C_2).
3. Simple passenger trains (C_3).
4. Freight trains (C_4).

Such categories are sorted with a decreasing order according to their priority. Special vehicles have then the highest priority and freight trains the lowest. Furthermore, if a passenger train has more passengers than another one, the cost of the delay will be more important. The objective function is then threefold:

- Scheduling trains according to their priority. For instance, a maintenance vehicle must be scheduled before a passenger train, if possible.
- Minimising the sum of delays.
- Maximising the overall passenger satisfaction. The passenger satisfaction decreases if its train is late.

A second objective function can then be expressed (Equation 13).

$$\texttt{lex\_min}\left(\sum_{t\in C_1} d_t, \sum_{t\in C_2} p_t \times d_t, \sum_{t\in C_3} p_t \times d_t, \sum_{t\in C_4} d_t\right) \tag{13}$$

Where p_t corresponds to the number of passengers of train t, as defined in Table 1. This equation gives a lexicographical ordering of trains according to their category from $C1$ to $C4$. For passenger categories (C_2 and C_3) the delay is expressed by passengers. In this way, the more is the number of passenger, the greater will be the penalty for delays. The objective is then to minimise this expression with regard to its lexicographical ordering.

Search Phase The exploration of the state space is performed with the algorithm of Vilim et al. [26] which combines a Failure-Directed Search with Large Neighborhood Search. The implementation proposed on CP Optimizer V12.6.3 [27] is particularly fitted to deal with conditional time-interval variables, precedence constraints, and optional resources. The search is performed on execution and on start date variables. Concerning the variable ordering, trains are sorted according to their category. For instance, activities related to passenger trains will be assigned before activities of freight trains. The value ordering is let by default.

4 Experiments

This section evaluates the performance of the Constraint Programming model through different experiments. Concretely, we compare our solution with the solutions obtained with classical dispatching methods on a realistic station: Courtrai. Its track layout is presented in Fig. 4. It contains 23 track segments, 14 signals, 68 itineraries and 84 possible routes. Three systematic strategies are commonly used in practice:

- **First Come First Served** (FCFS) strategy gives the priority to the trains according to their estimated arrival time.
- **Highest Delay First Served** (HDFS) strategy gives the priority to the train that has the earliest planned completion time.
- **Highest Priority First Served** (HPFS) strategy gives the priority to a train belonging to the category with the highest priority. The planned completion time is then used as a tie breaker for trains of a same category.

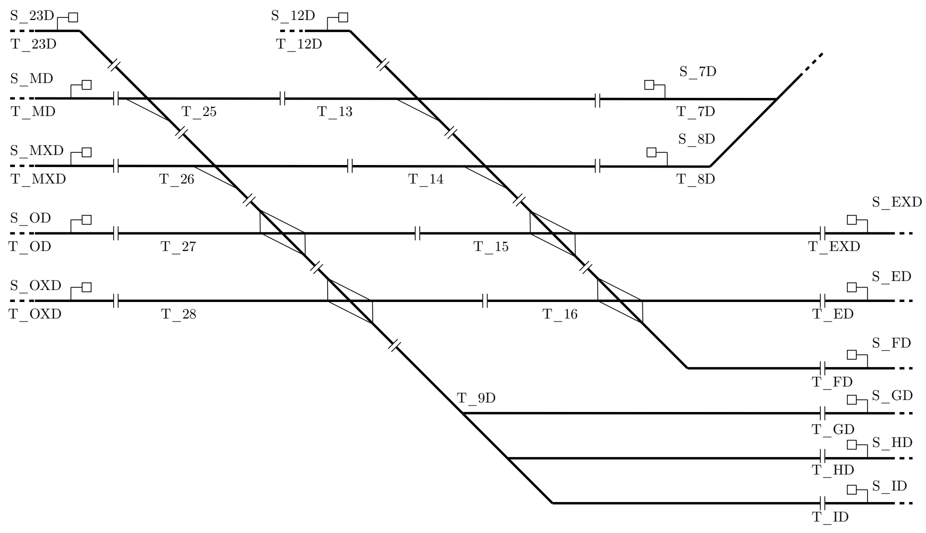


Fig. 4: Track layout of Courtrai.

Furthermore, three meta-parameters must be considered for the experiments, the *time horizon*, the *decision time* and *the number of trains*. The time horizon defines an upper bound on the estimated arrival time of trains. According to D'Ariano et al. [9], the practical time horizon for railway managers is usually less than one hour. In our experiments, we considered three time horizons (30 minutes, one hour and two hours). The decision time is the time that railway operators have at disposal for taking a decision. It is highly dependant to the criticality of the current situation. However, according to Rodriguez [23], the system must be able to provide an acceptable solution within 3 minutes for practical uses. Concerning the number of trains, we considered scenarios having 5, 10, 15, 20, 25 and 30 trains.

Experiments have be done under the assumption that we allow a feasible schedule. This responsibility is beyond the scope of the station. Two situations are considered: a homogeneous and a heterogeneous traffic.

All the experiments have been realised on a MacBook Pro with a 2.6 GHz Intel Core i5 processor and with a RAM of 16 Go 1600 MHz DDR3 using a 64-Bit HotSpot(TM) JVM 1.8 on El Capitan 10.11.15. The model has been designed with CP Optimizer V12.6.3 and the optimisation is performed with four workers.

Homogeneous traffic In this first situation, the number of passengers and the category are not considered. Each train has then the same priority and Equation 12 is the objective function used. Table 2 recaps the experiments performed.

Each scenario is repeated one hundred times with a random schedule. The different values of the schedule are generated randomly with uniform distributions. For instance, let us consider a schedule from 1pm to 3pm with 10 trains. For each train, we randomly choose its itinerary among the set of possible itineraries, its departure time and its expected arrival time in the interval [1pm, 3pm]. The delay indicated for each strategy corresponds to the arithmetic mean among all the tests. For each scenario, the average, the minimum and the maximum improvement ratio of the CP solution in comparison to the best solution obtained with classical methods is also indicated. POS indicates the number of tests where the CP approach has improved the solution while OPT indicates the number of tests where CP has reached the optimum.

As we can see, CP improves the solution for almost all the tests. The average improvement ratio is above 20% in all the scenarios. Optimum is often reached (more than 75 % of the instances) when 10 trains or less are considered.

Table 2: Comparison between CP and classical scheduling approaches for an homogeneous traffic with a decision time of 3 minutes.

Horizon	# trains	Average Delay (min.)			Improvement Ratio (%)			POS	OPT
		FCFS	HDFS	CP	Mean	Min	Max	(x/100)	(x/100)
2 hours	5	192.06	191.10	148.91	22.08	-7.69	100.00	99	97
	10	800.40	797.82	575.04	27.92	5.13	66.08	100	85
	15	1917.95	1896.64	1341.53	29.27	1.16	58.549	100	28
	20	3457.45	3397.03	2414.45	28.92	10.26	53.29	100	0
	25	5581.10	5632.69	3993.04	28.45	8.58	48.37	100	0
	30	8004.84	8018.76	5714.69	28.61	14.35	43.61	100	0
1 hour	5	225.36	228.75	172.06	23.65	0.71	100.00	100	96
	10	911.01	906.32	664.18	26.72	3.96	62.41	100	83
	15	2128.34	2104.95	1494.67	28.99	5.93	51.29	100	17
	20	3675.72	3680.00	2612.6	28.92	8.25	55.86	100	1
	25	5971.54	6004.81	4246.55	28.89	9.41	53.67	100	0
	30	8595.56	8576.92	6085.51	29.05	7.89	45.51	100	0
30 minutes	5	231.16	229.06	173.95	24.06	1.49	100.00	100	94
	10	950.30	929.71	692.18	25.55	3.12	64.32	100	83
	15	2145.36	2161.20	1535.86	28.41	7.927	51.61	100	14
	20	3858.67	3888.29	2728.0	29.30	6.63	52.50	100	0
	25	6137.02	6135.59	4320.14	29.59	7.75	53.38	100	0
	30	8863.49	8775.84	6357.08	27.56	8.72	49.08	100	0

Heterogeneous traffic In this second situation, we use Equation 13 for the objective function. The number of passengers and the category are then considered. As for the heterogeneous case, such values are chosen randomly with a uniform distribution. Among the classical approaches, only HPFS deals with an heterogeneous traffic. Table 3 recaps the experiments performed. Optimisations are performed sequentially for each category. The time is allocated according to the priority of categories. The allocation is then not done a priori but dynamically according to the time taken by the successive optimisations. Unlike the previous experiments, we do not compute the improvement ratio, but the number of experiences where CP has improved the solution obtained with HPFS. Our choice was motivated by the subjective aspect of defining an improvement ratio for a heterogeneous traffic. For instance there is no clear preference between decreasing the delay of one train of category C1 and of 10 trains with lower priorities. This kind of questions usually requires the consideration of signalling operators. We consider that CP has improved the solution when the sum of delay per category is lexicographically lower than the result provided by HDFS. For the three horizons considered and for 100 tests per scenario, CP improves the solution for almost all the tests, even when the optimum is not reached.

Table 3: Comparison between CP and HPFS approach for an heterogeneous traffic with a decision time of 3 minutes.

Horizon	# trains	POS ($x/100$)	OPT ($x/100$)
2 hours	5	100	99
	10	98	87
	15	100	64
	20	100	51
	25	100	13
	30	100	5
1 hour	5	100	99
	10	99	89
	15	100	75
	20	100	51
	25	100	22
	30	100	8
30 minutes	5	100	99
	10	99	87
	15	100	75
	20	100	52
	25	100	26
	30	100	7

Scalability This experiment deals with the scalability of the CP model in function with the decision time. We observed that setting the decision time to 10 minutes instead of 3 minutes do not increase significantly the performances. The gain of the improvement ratio is less than 1% for 60 random instances on an homogeneous traffic of 5,10,15,20,25 or 30 trains (10 instances per configuration). We can then conclude that even if the CP approach gives a feasible and competitive solution within 3 minutes, the quality of the solution do not increase significantly with time.

Criticality In some cases, railway operators do not have an available decision time of 180 seconds, they have to react almost instantly because of the criticality of the situation. For this reason, we analysed how the CP model performs with a decision time lower than 10 seconds. To do so, we recorded the number of experiments where the CP approach has improved the solution in comparison to FCFS and HDFS strategies. For the scenarios depicted in Table 2, we observed that CP provides a same or better solution in more than 99% of the cases and can then also be used to deal with critical situations.

Reproducibility A shortcoming in the literature about this field of research is the lack of reproducibility. To overcome this lack, we decided to provide information about our instances and the tests performed[1]. The information provided are enough to build a model, perform experiments and to compare them with ours.

5 Conclusion

Nowadays, railway operators must deal with the problem of rescheduling the railway traffic in case of real time perturbations in the network. However, the systematic and greedy strategies (FCFS, HDFS and HPFS) currently used for this purpose often give a suboptimal decision. In this paper, we presented a CP model for rescheduling the railway traffic on real time situations. The modelling is based on the recently introduced time-interval variables. Such a structure allows to design the model elegantly with variables and global constraints especially dedicated for scheduling. Finally, an objective function taking into account the heterogeneity of the traffic is presented. Experiments have shown that a dispatching better than the classical approaches is obtained in less than 3 minutes in almost all the situations that can occur in a large station, even when the optimum is not reached.

Two aspects are considered for our future works: improving the model and its scalability. Firstly, we plan to modify the model and analyse the consequences. For instance, we will use a `xor` instead of `alternative` constraint. Secondly, we will consider experiments on larger areas covering several stations.

Acknowledgments This research is financed by the Walloon Region as part of the Logistics in Wallonia competitiveness pole. Experiments have been performed with ILOG CP Optimizer through the academic initiative of IBM.

[1] Available at `https://bitbucket.org/qcappart/qcappart_opendata`

References

1. Higgins, A., Kozan, E., Ferreira, L.: Optimal scheduling of trains on a single line track. Transportation research part B: Methodological **30** (1996) 147–161
2. Ghoseiri, K., Szidarovszky, F., Asgharpour, M.J.: A multi-objective train scheduling model and solution. Transportation research part B: Methodological **38** (2004) 927–952
3. Zhou, X., Zhong, M.: Bicriteria train scheduling for high-speed passenger railroad planning applications. European Journal of Operational Research **167** (2005) 752–771
4. Harabor, D., Stuckey, P.J.: Rail capacity modelling with constraint programming. In: International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems, Springer (2016) 170–186
5. Senthooran, I., Wallace, M., De Koninck, L.: Freight train threading with different algorithms. In: International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems, Springer (2015) 393–409
6. Cacchiani, V., Huisman, D., Kidd, M., Kroon, L., Toth, P., Veelenturf, L., Wagenaar, J.: An overview of recovery models and algorithms for real-time railway rescheduling. Transportation Research Part B: Methodological **63** (2014) 15–37
7. Fay, A.: A fuzzy knowledge-based system for railway traffic control. Engineering Applications of Artificial Intelligence **13** (2000) 719–729
8. Higgins, A., Kozan, E., Ferreira, L.: Heuristic techniques for single line train scheduling. Journal of Heuristics **3** (1997) 43–62
9. Dariano, A., Pacciarelli, D., Pranzo, M.: A branch and bound algorithm for scheduling trains in a railway network. European Journal of Operational Research **183** (2007) 643–657
10. D'Ariano, A., Corman, F., Pacciarelli, D., Pranzo, M.: Reordering and local rerouting strategies to manage train traffic in real time. Transportation Science **42** (2008) 405–419
11. Schachtebeck, M., Schöbel, A.: To wait or not to waitand who goes first? delay management with priority decisions. Transportation Science **44** (2010) 307–321
12. Dollevoet, T., Huisman, D., Schmidt, M., Schöbel, A.: Delay management with rerouting of passengers. Transportation Science **46** (2012) 74–89
13. Dollevoet, T., Huisman, D., Kroon, L., Schmidt, M., Schöbel, A.: Delay management including capacities of stations. Transportation Science **49** (2014) 185–203
14. Corman, F., Goverde, R.M., DAriano, A.: Rescheduling dense train traffic over complex station interlocking areas. In: Robust and Online Large-Scale Optimization. Springer (2009) 369–386
15. Foglietta, S., Leo, G., Mannino, C., Perticaroli, P., Piacentini, M.: An optimized, automatic tms in operations in roma tiburtina and monfalcone stations. WIT Transactions on The Built Environment **135** (2014) 635–647
16. Araya, S., Abe, K., Fukumori, K.: An optimal rescheduling for online train traffic control in disturbed situations. In: Decision and Control, 1983. The 22nd IEEE Conference on, IEEE (1983) 489–494
17. Lamorgese, L., Mannino, C.: An exact decomposition approach for the real-time train dispatching problem. Operations Research **63** (2015) 48–64
18. Baptiste, P., Le Pape, C., Nuijten, W.: Constraint-based scheduling: applying constraint programming to scheduling problems. Volume 39. Springer Science & Business Media (2012)

19. Barták, R., Salido, M.A., Rossi, F.: New trends in constraint satisfaction, planning, and scheduling: a survey. The Knowledge Engineering Review **25** (2010) 249–279
20. Kelareva, E., Brand, S., Kilby, P., Thiébaux, S., Wallace, M., et al.: Cp and mip methods for ship scheduling with time-varying draft. In: ICAPS. (2012)
21. Kelareva, E., Tierney, K., Kilby, P.: Cp methods for scheduling and routing with time-dependent task costs. In: International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems, Springer (2013) 111–127
22. Ku, W.Y., Beck, J.C.: Revisiting off-the-shelf mixed integer programming and constraint programming models for job shop scheduling. Dept Mech. Ind. Eng., Univ. Toronto, Toronto, ON, Canada, Tech. Rep. MIE-OR-TR2014-01 (2014)
23. Rodriguez, J.: A constraint programming model for real-time train scheduling at junctions. Transportation Research Part B: Methodological **41** (2007) 231–245
24. Laborie, P., Rogerie, J.: Reasoning with conditional time-intervals. In: FLAIRS conference. (2008) 555–560
25. Laborie, P., Rogerie, J., Shaw, P., Vilím, P.: Reasoning with conditional time-intervals. part ii: An algebraical model for resources. In: FLAIRS conference. (2009)
26. Vilím, P., Laborie, P., Shaw, P.: Failure-directed search for constraint-based scheduling. In: International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems, Springer (2015) 437–453
27. Laborie, P.: Ibm ilog cp optimizer for detailed scheduling illustrated on three problems. In: International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems, Springer (2009) 148–162
28. Vilím, P.: Max energy filtering algorithm for discrete cumulative resources. In: International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems, Springer (2009) 294–308
29. Salido, M.A., Escamilla, J., Barber, F., Giret, A., Tang, D., Dai, M.: Energy-aware parameters in job-shop scheduling problems. In: GREEN-COPLAS 2013: IJCAI 2013 workshop on constraint reasoning, planning and scheduling problems for a sustainable future. (2013) 44–53
30. Hait, A., Artigues, C.: A hybrid cp/milp method for scheduling with energy costs. European Journal of Industrial Engineering **5** (2011) 471–489
31. Theeg, G.: Railway Signalling & Interlocking: International Compendium. Eurailpress (2009)
32. Cappart, Q., Schaus, P.: A dedicated algorithm for verification of interlocking systems. In: International Conference on Computer Safety, Reliability, and Security, Springer (2016) 76–87
33. Yamada, T., Nakano, R.: Job shop scheduling. IEE control Engineering series (1997) 134–134
34. Dechter, R., Meiri, I., Pearl, J.: Temporal constraint networks. Artificial intelligence **49** (1991) 61–95