A dynamic decomposition approach for the real-time Railway Traffic Management Problem

M. Petris[a,*], F. Naldini[b], P. Pellegrini[c] and R. Pesenti[d]

[a] Univ. Lille, Inria, CNRS, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France
matteo.petris@inria.fr

[b] COSYS-ESTAS, Univ Gustave Eiffel, IFSTTAR, Univ Lille, F-59650 Villeneuve d'Ascq, France
federico.naldini@univ-eiffel.fr

[c] COSYS-LEOST, Univ Gustave Eiffel, IFSTTAR, Univ Lille, F-59650 Villeneuve d'Ascq, France
paola.pellegrini@univ-eiffel.fr

[d] Department of Management, Università Ca' Foscari Venezia, Cannaregio 873, 30121 Venice, Italy
pesenti@unive.it

* Corresponding author

Extended abstract submitted for presentation at the 11th Triennial Symposium on Transportation Analysis conference (TRISTAN XI)
June 19-25, 2022, Mauritius Island

March 21, 2022

Keywords: railway traffic management, real-time scheduling, problem decomposition.

1 INTRODUCTION

Railway services are operated following a predefined timetable. However, their execution is often perturbed by unexpected events that make this timetable infeasible. Delay caused by these events is named primary delay, and it implies that trains occupy tracks at times that are different from the planned one. Depending on traffic and track layout, these late occupations may bring to *conflicts*, in which at least one train must slow down or even stop to preserve safe separation. This slowing down generates secondary delay, which may quickly propagate in the network.

Dispatchers can take actions to limit delay propagation, as train rerouting and rescheduling. They mostly do so manually. Several optimization approaches have been proposed in the literature to tackle this problem and support dispatchers (Cacchiani *et al.*, 2014). This problem is named real-time Railway Traffic Management Problem (rtRTMP). The great majority of the existing approaches either focus on geographically limited infrastructures represented microscopically (D'Ariano & Pranzo, 2008, Törnquist Krasemann, 2012, Pellegrini *et al.*, 2015), or merge some microscopic aspect to a macroscopic representation to deal with large infrastructures (Lamorgese & Mannino, 2015). One of the main challenges of the microscopic based approaches is their computational performance on large-scale networks. To deal with this challenge, few papers try to coordinate traffic management decisions made on several microscopic parts of infrastructures (Corman *et al.*, 2012), or on a somehow obtained decomposition of the overall problem. For example, (Luan *et al.*, 2020) presents various approaches to define decompositions a priori and to force the various traffic management decisions to be coherent. A rather different problem conception is proposed by (Van Thielen *et al.*, 2018): the decomposition of the problem changes with traffic evolution. Specifically, the paper defines an algorithm based on the so-called

dynamic impact zone. As soon as a conflict is detected, first rerouting possibilities for the two involved trains are assessed. If no rerouting possibility exists to eliminate conflicts, the further conflicts that will be generated by the possible resolutions of the considered one are identified and included in the dynamic impact zone. The conflict resolution is then decided by assessing its consequence in terms of delay propagation in the dynamic impact zone if first-come-first-served is applied.

In this work, we propose a neighborhood-based traffic management algorithm, following to some extent the problem conception of (Van Thielen *et al.*, 2018). We model the infrastructure microscopically. Our algorithm consists in making asynchronous traffic management decisions. Asynchronous decisions are made also in (Dal Sasso *et al.*, 2021), where train movements are split in so called temporal *ticks* to assess the presence of deadlocks in a single track network with passing loops. In our case, these decisions allow trains to reach their destination aiming at the minimization of delay propagation. Specifically, we identify the neighborhood of a reference train whenever a decision is to be made on its route or on the precedence with respect to another train. This neighborhood includes only the trains that may use track sections in common with the reference one, in the vicinity of the latter's current location and in the very near future. Traffic is then managed by applying an optimization approach as the one by (Pellegrini *et al.*, 2015), only considering trains in the neighborhood and the identified possibly common track sections. By doing so, alternative routes are taken into account, but they are as short as possible. This allows limiting the size of the instance and hence the optimization time. Moreover, whenever possible, it avoids making decisions that may have to be modified in the future, when the neighborhood is recomputed. However, routes must be long enough to guarantee that no deadlocks occur right out of the neighborhood due to the decisions made here. The algorithm starts identifying the concerned trains and their shortest but long enough routes. Then, it follows the principle that has proved to be successful in (Pellegrini *et al.*, 2015), simultaneously optimizing routes and schedules. We theoretically show that, in networks with some specific characteristics, this algorithm guarantees the achievement of a deadlock-free network-level solution, if it exists.

2 MODELING PRINCIPLES

We denote by BS and TC the set of block sections and track-circuits composing the infrastructure, respectively. We consider a set T of n trains traveling in the network. They may use different routes to reach their final destination: the set of the available routes for train t is denoted by $\mathcal{R}_t$. A train makes a decision upon alternative routes once it has reserved the block section where the switch that gives rise to the alternatives is located. We call *route decision block section* (RDBS) a block section where such decision is to be made: in a three-aspect system, it is the block section such that a further one exists between it and the block section including the switch. Given a route $r \in \mathcal{R}_t$, a *sub-route* $r_{bs \to bs'}$ of r is the sequence of block sections of r from bs to bs', where bs and bs' are block sections of r such that bs has to be traversed before bs' when traveling along r. To indicate that a sub-route is available for train t, we will write $r_{bs \to bs'} \subseteq r \in \mathcal{R}_t$. Moreover, given a subset of routes $\mathcal{R}' \in \mathcal{R}$ and a subset of block sections $BS' \in BS$, we denote by $Ext(\mathcal{R}', BS')$ the set of block sections composing sub-routes $r_{bs \to bs'} \subseteq r$, $r \in \mathcal{R}'$, where $bs \in BS'$ and $bs' \in BS \setminus BS'$ is the next RDBS on r. As an example, consider Figure 1: let $\mathcal{R}'$ contain routes r' and r'' and BS' the block sections up to the one identified by the pair of signals $s1$ and $s2$. Set $Ext(\mathcal{R}', BS')$ includes all the block sections along r' until the RDBS identified by $s3$ and $s4$ and those along r'' until the RDBS identified by $s5$ and $s6$.

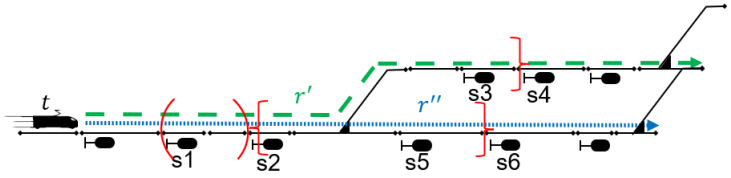

Figure 1 – *Graphical representation of relevant sets of block sections: round brackets show set BS', curly brackets indicate set $Ext(\mathcal{R}', BS')$ for train t having routes r' and r'' available.*

We describe railway traffic as a discrete event system. Events occur at the latest times at which route or precedence decisions must be made. They are the situations in which a train must exit a RDBS. Hereinafter, with a little abuse of notation, when we write *time* k we understand the time instant of the occurrence of event k. The *state* of the network at time $k = 0, 1, 2, \ldots$ is identified by: (i) the vector $p(k) = [p_t(k) : t \in T]$ of the positions of all trains, that is the last block section reserved by t; (ii) the set $Y(k) = \{y_{t,t',tc}(k) : t, t' \in T,\ tc \in TC\}$ of the precedences previously defined and, hence, in force between times $k-1$ and k, on common track-circuits. Specifically, a value $y_{t,t',tc} \in Y(k)$ is set equal to 1 (respectively to 0) if train t is planned to use tc before (respectively after) train t'. If no precedence has been fixed yet, $y_{t,t',tc}$ is undefined. At a certain time $\bar{k}$, the system is in its *final* state if all the trains reserved or crossed their final destination.

To conclude the section, let us introduce some sets of block sections that will be exploited in our algorithm (Section 3). Given the state of the system at time k, let $S_t(k)$ be the set of block sections that must be considered in the traffic management decisions involving train t. These are the block sections *claimed* by t at k. They are the ones that t may use between $p_t(k)$ and further RDBSs. Given $S_t(k)$ and a route $r \in \mathcal{R}_t$ such that $p_t(k) \in r$, we denote $\overline{bs}(r, S_t(k))$ the last block section in $S_t(k)$ that t encounters when traveling along r. We denote by $\mathcal{R}_t(k) = \{r \in \mathcal{R}_t : p_t(k) \in r\}$ the set of routes available for t at k and by $\overline{BS}(S_t(k))$ the set of all $\overline{bs}(r, S_t(k))$. For example, in Figure 1, position $p_t(k)$ of t is the block section identified by signals $s1$ and $s2$. Assuming $S_t(k)$ contains all block sections shown in the figure, including those derived by the extension discussed there, then $\mathcal{R}_t(k) = \{r', r''\}$. Here, $\overline{bs}(r', S_t(k))$ and $\overline{bs}(r'', S_t(k))$ are, respectively, the ones identified by pairs of signals $(s3, s4)$ and $(s5, s6)$.

3 SOLUTION ALGORITHM

The algorithm is based on a dynamic decomposition of the problem. Specifically, at time k, train t for which decisions are to be made is identified. Its neighborhood is defined and the rtRTMP is locally solved by iteratively calling the optimization algorithm RECIFE-MILP. The neighborhood constitutes a *sub-instance* of the overall problem. A sub-instance is defined by a pair $(Q_t(k), S(k))$, where $Q_t(k) \subseteq T$ is a subset of trains and $S(k) \subseteq BS$ is the subset of block sections claimed by the trains in $Q_t(k)$, i.e., those along which they may travel.

The procedure to determine a *short-term strategy*, i.e., a local solution to manage traffic when event k is triggered for train t is outlined in Algorithm 1. To lighten the notation, we drop the dependency on k when possible. In the initialization phase the algorithm takes a snapshop of the entire network. Given the positions of all trains $t' \in T$, it determines sets $S_{t'}$ of block sections claimed by t' (lines 2- 4). Subset of trains Q_t is set to be equal to the singleton containing t. Each iteration starts with a call to function SUB-INSTANCE_GENERATING_PROCEDURE, where some easy extensions of sub-instance (Q_t, S) are performed in order to avoid trivially infeasible iterations (line 7). For example, trains in $T \setminus Q_t$ sharing claimed block sections with trains in Q_t are included in Q_t. Then, we call RECIFE-MILP to determine a short-term strategy STS for sub-instance (Q_t, S) (line 8). The set of previously established precedences $Y(k)$ is also given as input to RECIFE-MILP. If a non-empty short-term strategy STS is found, the algorithm stops and returns STS (line 13). Otherwise, we try to further extend the sub-routes of some trains

Algorithm 1: SHORT-TERM_STRATEGY_GENERATING_PROCEDURE

Data: State $(p(k), Y(k))$ at time k, a train t
Result: STS
1 **begin**
2 **foreach** $t' \in T$ **do**
3 $S_{t'} \coloneqq \{p_{t'}(k)\}$;
4 **if** $p_{t'}(k)$ *is not RDBS* **then** $S_{t'} \leftarrow S_{t'} \cup Ext(\mathcal{R}_{t'}(k), \overline{BS}(S_{t'}))$;
5 set $Q_t \coloneqq \{t\}$, $S \coloneqq \bigcup_{t' \in Q_t} S_{t'}$ and $STS \coloneqq \emptyset$;
6 **while** $STS = \emptyset$ **do**
7 $(Q_t, S) \coloneqq$ SUB-INSTANCE_GENERATING_PROCEDURE(Q_t, S);
8 $STS \coloneqq$RECIFE-MILP$(Q_t, S, Y(k))$;
9 **if** $STS = \emptyset$ **then**
10 select $\tilde{Q} \subseteq Q_t$: $Ext(\mathcal{R}_{t'}(k), \overline{BS}(S_{t'})) \neq \emptyset \; \forall t \in \tilde{Q}$;
11 **if** $\tilde{Q} = \emptyset$ **then return** FAIL;
12 **foreach** $t' \in \tilde{Q}$ **do** $S_{t'} \leftarrow S_{t'} \cup Ext(\mathcal{R}_{t'}(k), \overline{BS}(S_{t'}))$;
13 **return** STS ;

(lines 10- 12). If an extension is possible we start the next iteration. Otherwise, the algorithm stops and we return failure. We observe that in the worst-case scenario the sub-instance may be extended to consider the entire network and all the trains travelling on it.

Finally, this algorithm enjoys an important property if the network of interest can be modeled as a series-parallel graph (Duffin, 1965):

Theorem 1 *The algorithm always allows to reach the final state of the railway traffic system on a network that can be modeled as a series-parallel graph, if at all possible.*

In the paper, we will formally prove the validity of this theorem and we will provide a proof-of-concept of the applicability of the algorithm.

References

Cacchiani, Valentina, Huisman, Dennis, Kidd, Martin, Kroon, Leo, Toth, Paolo, Veelenturf, Lucas, & Wagenaar, Joris. 2014. An overview of recovery models and algorithms for real-time railway rescheduling. *Transportation Research Part B: Methodological*, **63**(may), 15–37.

Corman, Francesco, D'Ariano, Andrea, Pacciarelli, Dario, & Pranzo, Marco. 2012. Optimal inter-area coordination of train rescheduling decisions. *Transportation Research Part E: Logistics and Transportation Review*, **48**(1), 71–88.

Dal Sasso, Veronica, Lamorgese, Leonardo, Mannino, Carlo, Onofri, Andrea, & Ventura, Paolo. 2021. The Tick Formulation for deadlock detection and avoidance in railways traffic control. *Journal of Rail Transport Planning & Management*, **17**, 100239.

D'Ariano, Andrea, & Pranzo, Marco. 2008. An Advanced Real-Time Train Dispatching System for Minimizing the Propagation of Delays in a Dispatching Area Under Severe Disturbances. *Networks and Spatial Economics*, **9**(1), 63–84.

Duffin, Richard J. 1965. Topology of series-parallel networks. *Journal of Mathematical Analysis and Applications*, **10**(2), 303–318.

Lamorgese, L., & Mannino, C. 2015. An Exact Decomposition Approach for the Real-Time Train Dispatching Problem. *Operations Research*, **63**(1), 48–64.

Luan, X., De Schutter, B., Meng, L., & Corman, F. 2020. Decomposition and distributed optimization of real-time traffic management for large-scale railway networks. *Transportation Research Part B: Methodological*, **141**, 72 – 97.

Pellegrini, Paola, Marliere, Gregory, Pesenti, Raffaele, & Rodriguez, Joaquin. 2015. RECIFE-MILP: An Effective MILP-Based Heuristic for the Real-Time Railway Traffic Management Problem. *IEEE Transactions on Intelligent Transportation Systems*, **16**(5), 2609–2619.

Törnquist Krasemann, Johanna. 2012. Design of an effective algorithm for fast response to the rescheduling of railway traffic during disturbances. *Transportation Research Part C: Emerging Technologies*, **20**(1), 62–78.

Van Thielen, S., Corman, F., & Vansteenwegen, P. 2018. Considering a dynamic impact zone for real-time railway traffic management. *Transportation Research Part B: Methodological*, **111**, 39 – 59.