Automatic generation of timetable with the ATMO tool

Technical Report

18/11/2021

Nicola Coviello, Giorgio Medeossi (Trenolab SRLS, Gorizia, Italy)
Thomas Nygreen (Jernbanedirektoratet, Oslo, Norway)
Paola Pellegrini, Joaquin Rodriguez (Université Gustave Eiffel, Lille, France)

trenolab

Trenolab SRLS, Via Maniacco 7/A, 34170 Gorizia, Italy
VAT N. IT 01166360311
info@trenolab.com

Document's Summary

1 Forewords

1.1 Introduction

This document presents a technical report of the project n°202000783 "*Tools for mathematical optimisation of strategic railway timetable models"* funded by the Norwegian Railway Directorate (Jernbanedirektoratet) and carried out by TrenoLab and Gustave Eiffel University.
To maximise the benefit of passengers and rail freight customers, the Norwegian Railway Directorate (Jernbanedirektoratet) and other railway agencies rely heavily on strategic timetables to identify necessary new or upgraded infrastructure and rolling stock. When converting a set of conceptual railway service requirements into a feasible timetable, the planners must balance several conflicting objectives, such as travel time, capacity utilisation and robustness. The timetabling process is time- and resource-consuming, and it often must be carried out for a significant number of different concepts, studying multiple alternatives for each concept. Automatisation (at least partial) of this process will therefore significantly improve planners' productivity in performing such tasks as capacity studies and strategic timetable planning.
The aim of the current research project is to develop a prototype tool to automatically generate timetable draft, called Multi-Objective Automatic Timetable Generator (ATMO). This document describes the algorithmic core of the tool, articulated into a Multi-Objective Ant Colony Optimisation (MOACO) algorithm and a Mixed Integer Linear Programming (MILP) formulation, the adopted data model for input and output, as well as a set of application case studies. The document is structured as follows: Section 1.3 presents a literature review about strategic timetabling and timetable-based railway capacity analysis, while Sections 2.1 and 2.2 describe the adopted data model and the relevant formulation for the TTP. Section 2.3 provides an overview on the adopted algorithmic framework, while Section 3 extensively describes the developed MOACO algorithm. The integration between MOACO and MILP formulation is addressed in Section 4. Numerical experiments are presented and discussed in Section 5 and conclusions are drawn in Section 6, proposing possible future developments of the tool and its algorithmic core. Section **Errore. L'origine riferimento non è stata trovata.** illustrates an implementation plan for the tool. Finally, Appendix A reports the specifications of the input/output data format.

1.2 Strategic timetabling and the TTP

Timetable planning requires looking at infrastructure capacity as a limited resource that can be exploited in different ways. Our framework is based on a multi-objective approach to provide the user with a set of timetables representing an approximation of a Pareto-Optimal Set (POS) in the objective functions (hyper-) space. The POS represents the best-found ways to exploit the available capacity, and is constituted by a set of non-dominated solutions, each solution being a timetable. The developed tool therefore implements this approach, thanks to an algorithmic core based on a novel multi-objective ant colony optimisation algorithm.
The Train Timetable Problem (TTP) is a classical problem in the field of Operations Research. It is notoriously NP-hard, meaning that for large instances exact methods likely fail to return the optimal solution in a reasonable time. No guarantee exists to find even high-quality feasible solutions quickly. Here, the size of practical interest instances is typically "large". Furthermore, exact methods as those based on an integer linear programming formulation tackled by commercial solvers cannot manage Paretian multi-objective optimisation within a single algorithm run. To this purpose, they need to be run repeatedly, involving significant time consumption.
Metaheuristics are algorithmic principles that can be instantiated to tackle virtually any optimisation problem. They have proven to be effective in tackling combinatorial NP-hard problems as the TTP. On one hand, they can provide rather "good" solutions within a reasonable computation time. Furthermore, they can be easily extended to perform multi-objective optimisation, and in particular to search for a POS of solutions. On the other hand, they often rely on sets of parameters that shall be carefully tuned, and they cannot ensure the solution's optimality.
In our approach, we exploit both the metaheuristic and exact perspectives, by combining a Multi Objective Ant Colony Optimisation algorithm and a Mixed Integer Linear Programming formulation tackled by a commercial solver. We adopt ACO for three main reasons. First, it suits well the resolution of the TTP since it builds solutions incrementally, thus mimicking real-practice timetables planning. This will foster the understanding and acceptance of the way the ATMO works by

practitioners. Second, it features effective exploration capabilities thanks to the blended exploitation of memory from past iterations (pheromone trails) and local information. On the one hand, pheromone trails can capture the implicit bonds between different choices eliminating the need for modelling them explicitly. On the other hand, local information can be provided by well-known timetable KPIs. Third, ants generate and maintain populations of solutions, which fits particularly well Paretian multi-objective optimisation. A consolidated literature background describes how to extend ACO to this purpose.

1.3 Bibliographical framework

1.3.1 Strategic timetabling and estimation of railway capacity

According to Abril et al (2008) strategic timetabling is a topic closely related to the estimation of railway capacity, the latter being estimated by means of the maximum number of trains which can be operated on a given railway infrastructure, within a certain time window and with given operative constraints. In practice, the capacity indicator shall be carefully formulated, since it strongly depends on several operative parameters like the traffic mix, the utilised rolling stock and the minimum level of service required by commercial needs (KFH Group, 2013). As a result, an absolute definition of railway capacity does not exist, as stated by the International Union of Railways: "The capacity of the railway infrastructure is not static, it depends on the way it is utilised" (UIC 2013). Because of such relativity, it is not possible to define a univocal method to evaluate railway capacity, instead a number of methods have been formulated in the last 50 years, each of them addressing a particular application range (see Kontaxi and Ricci, 2009).

Simulation-based methods represent a general-purpose approach and are not limited to particular application cases. They are normally based on an explicit model of the railway system, whose granularity can range between microscopic and macroscopic. The granularity of the model likely affects the overall accuracy, and can be used to scale studies based on a simulation approach. On one hand, simulations reproduce the railway traffic behaviour, considering several technical and operative constraints as well as stochastic phenomena as perturbations. On the other hand, the setup of these models (mainly if with microscopic granularity) requires a significant amount of input data, which, as highlighted by Liebchen and Schülldorf (2019), could not be always and completely available. These data are then handled by simulation algorithms, which can be affected by significant processing times and by the risk of not converging because of the occurrence of deadlock conditions. For instance, Khoshniyat and Törnquist Krasemann (2017) report some case studies in which convergence is not reached, highlighting how this could be a serious issue in practical applications.

Analytical methods provide a trade-off between model complexity, amount of required input data on one side and results accuracy and reliability on the other. The analytical method proposed by the UIC leaflet 406 (UIC, 2004) is adopted by several European Infrastructure Managers to calculate the capacity consumption of a given working timetable (Pouryousef and Lautala., 2015; Weik, et al., 2019). This method is of straightforward implementation with simple application cases, namely those of linear networks where full-line operation is prevalent (UIC, 2004 reports a typical application case).

Consolidated literature sources (Landex and Jensen, 2013; Lindner, 2011) agree upon the unsuitability of the UIC method to assess complex infrastructure topologies, like those of large stations or of highly interconnected railway nodes. Even the 2nd edition of the leaflet (UIC, 2013), which expands the capacity assessment of nodes, still presents recognised weaknesses (Weik, et al., 2019; Bešinović, 2018). In these cases, the reciprocal interdependence of several possible alternatives routes severely affects the simultaneous operation of trains in the same node area. Furthermore, additional technical and operative constraints could apply, which are not normally present in full-line operations. This behaviour cannot be properly grasped by analytical methods like the UIC one, thus resulting in a potential inaccuracy of the results.

Hansen (2000) points out how the application of analytical methods to large stations or nodes shall be supported by further empirical considerations drawn from real traffic data. In these cases, simulations can be used as a validation of the results.

In general, simulation methods represent the best and possibly the only universally valid approach to consistently analyse the capacity of complex railway nodes. Several simulation methods are available, most of them embedded in calculation packages under commercial licensing. In general, these packages are conceived and designed with synchronous simulation in mind. On the one hand they provide several and powerful functionalities for simulating and analysing a given timetable

(assumed as feasible), possibly considering traffic perturbations. On the other side, they require a timetable to be entered as an input, and provide rather a "validation" of this timetable. Pouryousef et al. (2015) provide two extensive and comprehensive reviews about these tools. Simulation tools do not normally include tools for the automatic generation of feasible timetables; even if some remarkable exceptions can be found. Such functionalities are however normally limited to automatic fine-grane refinements of timetables, as the so-called conflict solving: given an already arranged timetable with residual conflicts, the tool removes them varying paths in a neighbourhood of the previded ones. Relevant literature can be found in Weymann and Nießen (2015) and Maurer et al. (2021)

1.3.2 The Train Timetabling Problem

Railway traffic simulation models can be used to arrange input datasets for the resolution of the Train Timetabling Problem (TTP, Hansen and Pachl, 2014). This classical problem in the field of operation research consists in defining the arrival and departure times of trains in stations and in selecting their routing across the network. In the bargain, a solution of the TTP is normally represented by a timetable which is optimal according to one or more objective functions (e.g., minimisation of the total travel time, minimisation of the total connection time, etc.) and which respects a set of constraints. In some models, a controlled violation of certain constraints is accepted. Constraints can be roughly classified into technical and operative ones. The firsts make the resulting timetable feasible, i.e. ensure that in real operation trains can respect their schedule, at least in absence of traffic perturbations (Goverde and Hansen, 2013). Operative constraints foster the compliance with commercial or organisational needs (passenger connections and transfer times, crew and rolling stock rostering, etc.).
Several approaches have been pursued to solve the TTP, as Mixed Integer Linear Programming (MILP) formulations or meta-heuristic techniques. Cacchiani et al. (2016) provide an extensive and up-to-date review about this topic, while Caimi et al. (2017) describe various modelling and solution methods for solving railway timetabling problems. In the remaining of this Section some examples of projects and software tools used for the automatic generation of feasible timetables are presented and discussed.
A particular declination of the TTP is the railway timetable saturation problem (TSP), consisting in the generation of feasible timetables which exploit all the available capacity (Delorme et al., 2001). This is normally accomplished by maximising the number of scheduled paths, i.e. inserting additional courses into a given timetable while respecting given technical and operative constraints. Even if this is the common acceptation of the concept of timetable saturation, we should remark that all the available capacity could be exploited also without inserting any additional path, for instance by re-arranging an existing timetable inserting extra stability margins. With this meaning, a timetable is saturated when all available capacity is used.
By analysing a timetable assumed as saturated through a proper set of KPIs it is possible to get a numerical quantification of the capacity of the system. In this way a saturated timetable depends on both technical constraints (which are considered invariant) and operative ones. The latter would define just one of the several different ways in which the system can be operated. As a result, capacity shall not be represented by a set of punctual values of the aforementioned KPIs, but rather by their variation ranges. A review of the saturation method proposed so far is presented in Coviello et al. (2017). Two of the most recent contributions are those by Pellegrini et al. (2017) and by Peterson et al. (2019). The first presents a saturation method based on a MILP formulation which produces saturated timetables with guaranteed optimality (provided that the computation time does not exceed a given upper bound). This formulation relies on a microscopic model of railway operations, directly obtainable from microsimulations, with a high accuracy of the relevant results. The model is solved with the commercial solver CPLEX. Peterson et al. (2019) tackle the saturation problem using an extended algorithm for thick paths search in polygonal domains. This algorithm provides optimal solutions, but relies on a rather aggregate data model which necessarily entails a loss of accuracy.
Strategic timetabling requires an approach similar to timetable-based capacity analysis. In early-stage timetable planning, timetable requirements (and operation constraints) are normally only partially defined, meaning that a variety of optimal solutions can be designed to fulfil them. The task is therefore to provide the planner with the whole set of optimal timetables, each of them qualified by the values of the considered KPIs.

The open railway market is encouraging European Infrastructure Managers to carefully define and quantify the capacity of the relevant railway systems, together with the possible optimal ways to fully exploit it (Schupbach et al., 2017). Railway Operators, who are confronting themselves with a growing competition, apply for more capacity both in terms of number of train paths and of their commercial requirements (Broman, Eliasson, & Aronsson, 2019). The application of advanced methods for evaluating capacity represents a key action for a better exploitation of railway networks. Nowadays, significant research efforts are being carried on in order to improve the methods and the algorithms devoted to the solution of the TTP. Main current issues concern their actual applicability to real case studies, characterised by big-sized problem instances. For instance, Jordi et al. (2019) experiment different approaches proposed in literature to test their quality and performance. The aim of this activity is to find the best solution to implement an integrated capacity planning tool for the Swiss Railways. The study highlights how the capability to solve big-sized problem instance (e.g., the whole Swiss network) is one of the main problems encountered. Other major issues are reported by Bešinović (2018), which describes how existing integrated timetabling approaches lack in efficiency, stability, feasibility or robustness of solutions. In general, as stated by Lamorgese (2017) exact resolution of TTP models is often impossible, and even to found feasible solutions in a reasonable time is a difficult task. To this purpose, decomposition techniques in combination with heuristics are often adopted, as in Goverde et al. (2016) and Lamorgese et al. (2017). This is also the approach followed in the present research, where a metaheuristic MOACO algorithm provides a fast-but-coarse exploration of the solutions' space, and a MILP formulation performs a further refinement of solutions.

2 Model and algorithm

2.1 Infrastructure data model

The ATMO transforms a so-called service concept into a working, feasible timetables. A service concept is a set of conceptual railway service requirements and it's basically a raw timetable draft which provides the number and type of trains that have to figure in the working timetables, respecting given technical and operational constraints.

We adopt a macroscopic model for infrastructure and operations, based on a multigraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Nodes represent timing locations (stations, junctions, halts) in the rail network. For each node $\ell \in \mathcal{V}$ we define:

- A set of tracks $t \in \mathcal{T}_\ell$, one of which is identified as the main one;
- An additional running time Δrt_{rs}^{t} if a track t different from the main one is used by a rolling stock rs (0 in case of using the main track);
- For any each track, a minimum separation time between the end of the occupation by a train and the beginning of the occupation by the following one tg_ℓ, providing that each track can be used by at most one train at time.

Edges are line stretches connecting consecutive timing locations. Edges can be mono- or bi-directional, and more than one edge can connect the same pair of consecutive stations. Any edge $e \in \mathcal{E}$ is represented by $e = (\underline{e}, \overline{e}, j)$, where the origin and destination of e are in the set of nodes ($\underline{e}, \overline{e} \in \mathcal{E}$) and j is an index in $[1, J_e]$ to distinguish different edges connecting the same pair of nodes (for example to represent the presence of multiple-track line). A set of rolling stock types $rs \in RS$ can travel on a line stretch. A rolling stock cross each location in two different modes $\gamma \in \{PASS, STOP\}$ (passing or stopping). Therefore, four type of events can characterise the traveling along a line stretch, depending on the so-called edge extremity events (EEE). These can be pass/pass, pass/stop, stop/pass, stop/stop: $\varepsilon \in EEE$ is a pair indicating what event occurs at the origin and at the destination node, respectively.

On an edge $e \in \mathcal{E}$ we have:

- A technical minimum run time $mrt_{e,dir,\varepsilon}^{rs}$ depending on the rolling stock of the train ($rs \in RS$), on the direction it is travelling into (dir) and on the EEE chosen ($\varepsilon \in EEE$). This run time includes the running time in both extreme nodes if the main track is used;
- A maximum energy consumption corresponding to the minimum technical run time $mec_{e,dir,\varepsilon}^{rs}$ depending on the rolling stock of the train ($rs \in RS$), on the direction it is travelling into (dir) and on the EEE chosen ($\varepsilon \in EEE$);
- A minimum headway $mh_{e,dir,\varepsilon_1,\varepsilon_2}^{rs_1,rs_2}$ between the entrance of pairs of trains running in the same direction dir. It depends on the rolling stock of the two trains (rs_1 and rs_2

respectively) and of the EEE of each train ($\varepsilon_1, \varepsilon_2 \in EEE$). The value of $mh^{rs_1,rs_2}_{e,dir,\varepsilon_1,\varepsilon_2}$ is to be considered if the train with rolling stock rs_1 passes first.

- For trains running in opposite direction on a bi-directional edge (single track line), the minimum headway is imposed at the passage at the same extremity of the edge, which for one train will be the origin and for the other the destination. It depends on the direction dir of the train passing first: it is $mh_{e,dir,\underline{e}}$ for the origin node along this direction, and $mh_{e,dir,\overline{e}}$ for the destination.

In our model trains can be scheduled with running times greater than the minimum technical ones. We adopt a linear relationship to model the relevant variation of energy consumptions and headways between trains running into the same direction. If for a given rolling stock $rs \in RS$ a run time $rt > mrt^{rs}_{e,dir,\varepsilon}$ is chosen, the energy consumption is reduced by $kec^{rs}_{e,dir} \cdot \left(rt - mrt^{rs}_{e,dir,\varepsilon}\right)$, i.e.

$$ec = mec^{rs}_{e,dir,\varepsilon} - kec^{rs}_{e,dir} \cdot \left(rt - mrt^{rs}_{e,dir,\varepsilon}\right) \quad \text{Eq. 1}$$

For a pair of trains t_1 and t_2 (using rs_1 and rs_2 respectively), if the running time of the first train is higher than the minimum one, the headway is increased by $kh^{rs_1}_{e,dir} \cdot \left(rt_1 - mrt^{rs_1}_{e,dir,\varepsilon}\right)$, i.e.

$$h^{t_1,t_2} = mh^{rs_1,rs_2}_{e,dir,\varepsilon_1,\varepsilon_2} + kh^{rs_1}_{e,dir} \cdot \left(rt_1 - mrt^{rs_1}_{e,dir,\varepsilon}\right) \quad \text{Eq. 2}$$

2.2 Timetable data model

Timetable input data describe the train services that shall be scheduled in the resulting timetables. This information is provided for each train group, being a train group set of periodic trains. Spare courses are modelled as single-train groups. Three types of groups can be defined: fixed groups represent a mere constraint to the timetabling process; movable group can be adjusted in time and space (station routing) to optimise the resulting timetables; optional group can also be de-activated by the algorithm. Priority is defined by means of a priority factor which weights the objective functions' values. As it will be explained in Section 3, the MOACO algorithm works with discretized time. To this purpose, time discretization is defined for each group. This allows to perform a "fine" timetabling for certain groups only, and a "coarser" one for the remaining groups.

Each group g is characterised by:

- The type $t_g \in \{FIXED, MOVABLE, OPTIONAL\}$;
- The period p_g;
- The number of trains belonging to the group nt_g;
- The priority factor $pr_g \geq 1$;
- The journey J_g, defined as the list of the pair of directed edges used by the group's trains. We define with $\underline{l}$ and $\overline{l}$ the first and last locations (nodes) in J_g respectively;
- The time discretization φ_g.

For each location (node) l visited along the journey J_g, the following constraints are defined by the service concept:

- The minimum and maximum arrival times at l, $\underline{arr}_{g,l}$ and $\overline{arr}_{g,l}$ respectively. They are defined, for the first train of the group, only if $l \neq \underline{l}$.
- The minimum and maximum departure times from l, $\underline{dep}_{g,l}$ and $\overline{dep}_{g,l}$ respectively. They are defined, for the first train of the group, only if $l \neq \overline{l}$.
- The rolling stock used by the group's trains on the edge leaving l, only if $l \neq \overline{l}$;
- The set $\mathrm{T}_{g,l}$ of usable station tracks;
- The set $\Gamma_{g,l}$ of usable pass/stop modes;
- The minimum and maximum stop times $\underline{stop}_{g,l}$ and $\overline{stop}_{g,l}$. Remark that is there is no mandatory stop at l, we will have $\underline{stop}_{g,l} = 0$ and $\overline{stop}_{g,l}$ large constant. If stops are forbidden, then $\underline{stop}_{g,l} = \overline{stop}_{g,l} = 0$;
- The minimum and maximum run times admitted for the group's trains in the edge e leaving l (only if $l \neq \overline{l}$), $\underline{rt}_{g,e}$ and $\overline{rt}_{g,e}$;
- The so-called periodicity tolerance $tol_{g,l} \geq 0$, defined as the maximum deviation (in absolute value) from a strictly periodic pattern admitted for the group's trains in that

location. A tolerance set to 0 imposes that, in that location, arrivals and departures different the trains of the group are separated by exactly a multiple of the group's period p_g.

2.3 ATMO architecture

The ATMO tool developed in this research uses a multi-objective approach to identify timetables that best utilise available capacity to operate a specified traffic pattern. The tool exploits an algorithmic framework based on a novel Multi-Objective Ant Colony Optimisation (MOACO) algorithm and on a Mixed Integer Linear Programming (MILP) formulation.

The MOACO algorithm quickly performs a wide-ranging exploration of the possible solution space. Timetables found by the MOACO are then refined using a MILP formulation. More specifically, a commercial MILP solver is used to identify possible local improvements to the timetables. Although ACO algorithms have already been applied in the train routing problem (Samà et al., 2016), the algorithmic framework proposed in this research is, to the best of our knowledge, a novel and original contribution in the field of mathematical optimisation applied to the entire routing-scheduling problem.

The combination of MOACO and MILP takes optimal advantage of each method's strengths providing users with the Pareto Optimal Set (POS) of possible timetables in a reasonable computation time. Providing a set of timetables rather than single timetables is especially useful in the strategic timetable and network capacity planning, where comparison between alternatives is a crucial step.

Finally, the Paretian multi-objective approach has a major practical advantage over methods where multiple objectives are blended together by weights in order to get a single objective function to be optimised. The latter are often significantly sensitive to the weights' values, thus requiring a careful tuning to obtain sound results. Furthermore, estimates of the monetary value of units of each of our objectives can be affected by (also quite significant) error margins.

The algorithmic framework generates timetables while attempting to optimise them according to the following five objectives:

- TTT: Minimise total travel time of all trains.
- EC: Minimise total energy consumption.
- ST: Maximise timetable stability.
- NTR: Maximise the number of optional trains that can be scheduled.
- CFL: Minimise the number of residual traffic conflicts (optional, user can also choose to be provided with conflict-free timetables only).

These objectives are computed by means of specific KPIs, calculated on the resulting timetables:

1. Travel times are trivially calculated as the difference between the arrival time at a train's last station and the departure time from the train's first station. The total travel time is the weighted sum of the trains' travel times, weights being given by groups' priority factors.
2. Energy consumption strictly depends on the running times of trains in each infrastructure edges, as well as on whether trains perform passes or stops in stations where both options are allowed. All energy consumption values are input data. The total energy consumption is the weighted sum of the trains' energy consumptions, weights being given by groups' priority factors.
3. The total weighted number of optional trains scheduled is trivially computed, multiplying the number of trains of each optional group by its priority factor.
4. Stability can be evaluated by means of several KPIs (Goverde and Hansen, 2013). For a strong integration within the ACO architecture, we consider a KPI that can be incrementally computed during the solution construction. Specifically, we maximise the minimum buffer time in the timetable. A buffer time is the time separation between two feasible consecutive utilisations of the same infrastructure resource (line stretch or station track) minus the minimum separation imposed between them (minimum technical headway).

This multi-objective approach makes the ATMO especially useful for strategic planning because it supports planners in solving several types of problems including saturation studies, strategic exploration of draft timetables, fine-tuning of previously developed timetables, and capacity assessment of infrastructure alternatives. New objectives can be added, or substituted for the ones we have used, provided that the associated KPI can be calculated efficiently. For objectives with

several possible KPIs, the choice may depend on the application. One could even include more than one KPI for the same objective, e.g., to measure stability under different perspectives.

Strategic planning consists of analysing and comparing various combinations of infrastructure and timetable concepts. Therefore, several different infrastructure models and timetables are entered into the tool. Once all the input data has been entered, the planner chooses which objective functions should be considered in developing the optimised timetables. Finally, the tool is used to create many timetables that meet the selected criteria.

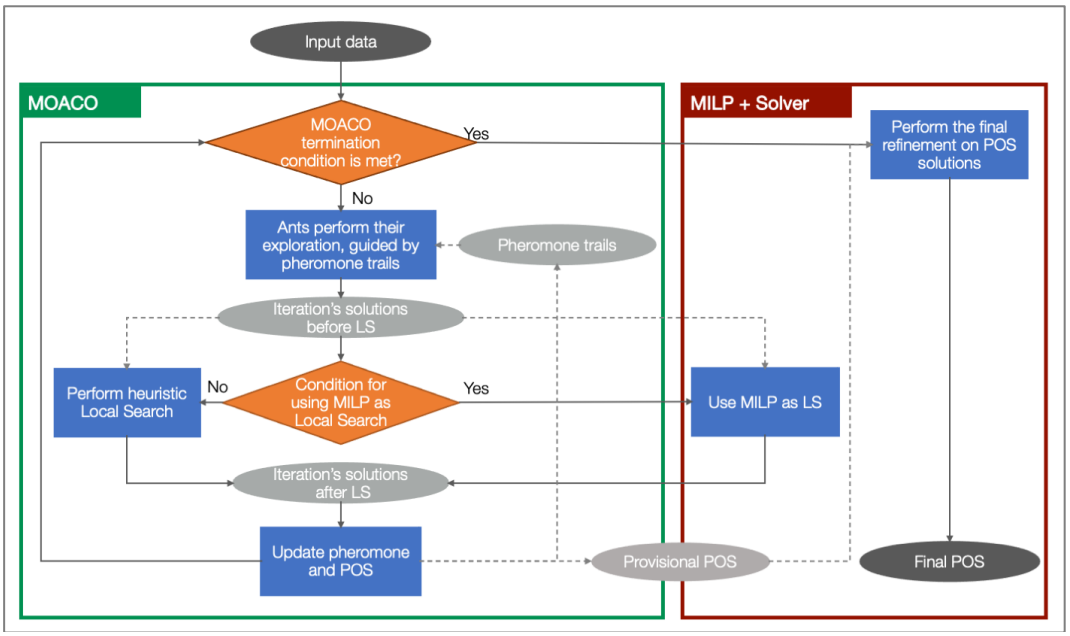

Figure 1. Global architecture of the algorithmic framework. Grey ovals represent data contents, blue rectangles processes and orange diamonds conditional switches. Solid arrows show the operations flows, dashed arrows show data flows.

Figure 1 displays the architecture of the algorithm. Grey ovals represent data contents, blue rectangles processes and orange diamonds conditional switches. Solid arrows mark the operations flows, while dashed ones are data feedings. Starting from input data, the Multi-Objective ACO (MOACO) through multiple iterations maintains and updates a provisional POS. The best-so-far solutions in this POS guide the ACO search during subsequent iterations via the pheromone trails. The MILP formulation further improves these solutions. It plays a double role: first, it acts as a Local Search which refines some of the POS solutions during the MOACO search process. Alternatively, local search is performed by a simpler heuristic. The criterion for selecting the local search procedure is defined by the user. For example, MILP local search can be performed once every ten iterations. We anticipate here that the usage of MILP as a Local Search is only proposed in this paper, leaving its calibration and analysis to further research. Second, after the MOACO algorithm stops according to a termination criterion, the MILP formulation refines all the solutions in the POS before their presentation to the user.

The MILP formulation deals with a model that is slightly different from the one considered by MOACO.

On the one hand, conflict constraints (both in line stretches and in station tracks) are relaxed within the MOACO algorithm to improve its searching capability. In particular, MOACO uses an additional objective, i.e., the minimisation of conflicts. As it is explained in Section 3, priority is accorded to conflict minimisation with respect to the other objectives. Yet, some residual conflicts may remain in the MOACO timetables. They are solved (if possible) in the MILP stage, which considers conflict constraints as hard ones.

On the other hand, MOACO enforces rigid regular intervals between trains of the same train group. This permits to dramatically speed-up the exploration. However, it makes it impossible to profit of periodicity tolerances (defined as input data) to solve conflicts and improve objective function values. In the MILP formulation, train paths are free to vary exploiting the abovementioned tolerances. Periodicity tolerance is a useful parameter for strategic planning. On one hand, strict periodicity is more attractive for passengers because precise clock-face schedules are easier to

remember. On the other, strict periodicity consumes extra capacity and therefore reduces the number of trains that can be operated especially for single track lines (Emery, 2010 and Coviello, 2014).

3 A MOACO for the TTP

3.1 Fundamentals of ACO

Ant Colony Optimisation algorithms, first proposed by Dorigo in 1992, are based on the construction of feasible solutions performed by a set of agents (ants) during a certain number of iterations. At a given iteration, each ant builds its solution independently of the other ants. The construction of a feasible solution consists in the sequential selection of a set of nodes on a pre-defined construction graph. Each node of this graph is therefore referred to as a *solution component.* This approach suites directly discrete combinatorial problems, where a solution is naturally composed by a set of discrete components. In case of continuous problem (as the scheduling ones, where the time variable is likely to be considered), a discretization is required. In our approach, discretization is performed when the Time Expanded Graphs are defined (see Sections 3.2 and 3.6.1).
In general, a solution can be a *path* or a *clique* on the construction graph, depending of the so-called *pheromonal strategy (PS).* Given a partial solution (i.e. a set of components already selected), the choice of the next component to be added is performed within a set of *candidates*.
The candidates are defined according to the pheromonal strategy (Solnon and Bridge, 2006):

- In case of *clique PS,* the candidates are those nodes for which a linking edge exists with all the components already selected;
- In case of *path PS*, the candidates are the neighbours of the last component of the partial solution.

Each candidate is chosen according to the pseudo-random proportional rule, i.e. with a probability given by the weighted product of two factors τ and η, called pheromonal and heuristic factors respectively.
The pheromonal factor depends on a quantity, the virtual pheromone, which has been laid on the construction graph's edges by some ants during previous iterations of the algorithm. Given that ants that laid pheromone were the best-performing ones in the relevant iterations (i.e. those which produced the better solutions), the pheromone represents a "memory" of the components that more frequently contributed to produce good solutions. On the other side, at each iteration the pheromone of all edges evaporates, i.e. decreases of a certain amount. In this way edges that are no more producing good solutions are more likely to be neglected. Evaporation and increment of the edges' pheromone are performed at the end of each iteration, during the *pheromone update step*.
The heuristic factor provides a local measure of the desirability of a candidate. It can be computed *statically* or *dynamically*. In case of static computation, it actually depends on an a-priori attribute of each edge. This is the more computationally efficient implementation, since the heuristic factor is calculated only once, during the definition of the construction graph. In case of dynamic computation, the heuristic factor depends on the partial solution built so-far, and therefore it has to be calculated each time a candidate is to be selected.
At the end of each iteration, during the *pheromone update step* a set of ants is allowed to increment the pheromone on the edges of the respective solution. The definition of this set of ants is a crucial step. With non-elitist approaches, only the best-performing ants of the current iteration are allowed to lay pheromone. With elitist approaches, it is maintained a record of the best performing ants so-far, and only a subset of them would contribute to update pheromone.
In the following sections we will describe how this basis architecture:

- Is developed onto two layer in order to tackle the TTP;
- Is properly integrated in order to perform Multi-Objective optimisation.

3.2 A two-layers architecture

In this section, we provide an overview of our original Ant Colony Optimisation algorithm for the multi-objective automatic railway timetable generation. The proposed ACO is a multi-objective extension of the Max-Min Ant System algorithm. The designed ACO extension follows the guidelines proposed by López-Ibáñez and Stützle (2012) for multi-objective optimisation.
In MOACO, we define a two-layer architecture. It applies to the TTP an approach already used for ACO applied to the Multi-Depot Vehicle Routing Problem (Yao et al. (2014)) and to the Course Timetabling Problem (Nothegger et al. (2012)). This two-layer architecture actually mimics the real-

world timetabling procedure, performed mainly by hand by specialized timetable planners. Two basic actions can be identified in this procedure:

A1. The selection of the next train group to be scheduled. Here we also include the decision whether to schedule or not an optional train. Two main criteria may guide this choice:
    a. The priority of each train group, with respect to the other ones. In principle a higher priority train should follow its ideal timetable path more than a lower priority one. A train scheduled before another one is likely to be designed closer to its ideal path since it is subject to fewer constraints.
    b. In case of optional train groups, the estimation of existing conflicts with already scheduled ones. This may lead to the decision of not-scheduling a train group in case it is believed that it would not fit into an already populated timetable.

A2. The scheduling of a certain train group within a timetable already populated by previously set train-paths. The scheduling will actually define the arrival and departure times, as well as the utilised station track, of the train group in each station of its journey.

In principle, an action A1 (selection of the next train group to be scheduled) is always followed by an action A2 (scheduling of the selected train group). A2 is skipped when A1 chooses not to schedule a certain train group. In reality, timetable planners iteratively repeat pairs of A1-A2 actions.

Exploiting this architecture, we define two types of construction graph to be explored by artificial ants in MOACO.

A Layer 1 graph is associated to actions of type A1. They concern the decision of which group of trains actually figure in the timetable and in which order they are scheduled.

A set of Layer 2 graphs is associated to actions of type A2. These are *time-expanded directed graphs* (TEG). They refer to path schedule decisions, considering one group of trains in each graph, as proposed by De Fabris et al. (2014).

Our ACO algorithm exploits this structure. We consider that a TTP solution (i.e. a railway timetable) is obtained by the combination of the following sub-solutions (SS):

- A Layer 1 sub-solution $S1$, which models the sequence of actions A1, defining:
    - The trains actually scheduled in the resulting timetable;
    - The order in which these trains are scheduled.
- A set of Layer 2 sub-solutions $S2_g$, one for each scheduled train group g. Each of them describes how a train is actually scheduled, in terms of arrival/departure times, pass/stop event and used track in each station.

A solution S is therefore defined by

$$S = \left\{S1, S2 = \left\{S2_g, for\ each\ scheduled\ group\ g\right\}\right\}$$

Each SS is constructed by the virtual ants by exploring a dedicated *construction graph*, with its own pheromone trails and heuristic information. Specifically, each ant starts choosing a Layer 1 node, then, if this entails to schedule a group, it builds a path on the corresponding Layer 2 graph. Then the ant chooses the next Layer 1 node and the process isiterated. During the construction of the TEG path, interactions with already-built L2 solutions are taken into account This procedure continues until decisions are made for all groups at Layer 1.

We model our MOACO after Max-Min Ant Systems (MMAS, Stützle and Hoos, 1997, 2000), an ACO variant which proved effective for a large number of different combinatorial problem. We implement a clique pheromonal strategy for Layer 1 (Solnon and Bridge, 2006), and a more classical path strategy for Layer 2. To the best of our knowledge, this hybridisation is a novel contribution to the field of ACO.

The edges of the construction graphs are used to store pheromone. Each time a solution will be used to update the pheromone, it will be incremented by a certain δ on the edges of Layer 1 and Layer 2 construction graphs used by that solution.

3.3 Performing multi-objective optimisation

We extend the MMAS approach to the multi-objective variant following the multi-colony architecture proposed by López-Ibáñez and Stützle (2012). In this section we describe the main points constituting such an architecture, which will then be recalled in following sections.

3.3.1 Aggregation

In a multi-colony architecture, ants are divided into subsets called *colonies*. Each colony features its own pheromone storage structures (one for each objective) and sets of aggregation weights.

Aggregation weights, defined in the interval [0,1] are used to blend together the pheromonal and heuristic information relevant to different objectives during the ants' search. The blending is performed through the linear combination of the values relevant to different objectives.
We define the set of objectives

$$O = \{TTT, EC, NTR, ST, CFL\} \quad \text{Eq. 3}$$

and a colony $\mathbb{C}_o$ for each objective $o \in O$, on whose optimisation the colony's ants will mostly focus. We call o the *colony's main objective*. For each colony $\mathbb{C}_o$, we define a set Λ_o of $N^{weights}$ aggregation weights for the main objective as follows

$$\Lambda_o = \left\{\lambda_o^i = 1 - \left(1 - \underline{\lambda}\right) \cdot \frac{i-1}{N^{weights} - 1}, i = 1, \dots, N^{weights}\right\} \quad \text{Eq. 4}$$

$\underline{\lambda} \in [0,1]$ being the minimum value in Λ_o. Weights in Λ_o are those used by colony $\mathbb{C}_o$ for aggregating the objective o during $N^{weights}$ consecutive iterations. If at iteration $iter$ the colony uses the weight $\lambda_o^{N^{weights}}$, at iteration $iter + 1$ it will use the weight λ_o^1. At each iteration $iter$, the weights for objectives $\hat{o} \in O - \{o\}$ (different from the colony's main objective) are randomly chosen in such a way that

$$\lambda_o^i + \sum_{\hat{o} \in O - \{o\}} \lambda_{\hat{o}}^i = 1, with\ i = i(iter) \quad \text{Eq. 5}$$

This defines the set of weights Λ_o^{iter} used by that colony in that iteration.
It may be pointed out that, according to this implementation, the higher $\underline{\lambda}$ is, the more the search of the colony $\mathbb{C}_o$ is pushed towards the optimisation of its main objective, neglecting the other ones.

3.3.2 Normalisation

During aggregation, heuristic information relevant to different objectives is blended by means of the aggregation weights. This information is, in general, dimensionally different, and depends to each objective to be evaluated. For example, in our application we have time for TTT, energy for EC, units for NTR, time for ST and units for CFL. In general, also if heuristic information was dimensional consistent, their values may differ also of orders of magnitude.
For these reasons, we introduce a sation step before the aggregating heuristic information. To this purpose, we take advantage of the main feature of the MMAS implementation, which limits the pheromone's values between a lower and an upper bound, τ_{MIN} and τ_{MAX} respectively. The normalisation operator for a variable x is defined as

$$norm(x, best, worst) = \tau_{MAX} - \frac{best - x}{best - worst} \cdot (\tau_{MAX} - \tau_{MIN}) \quad \text{Eq. 6}$$

Given a variable x, and its possible best and the worst values ever, the operator linearly interpolates x between the two bounds. It is worthwhile to highlight that for the operator it is indifferent whether $best > worst$ or $best < worst$. Therefore, it neglects the optimisation sense (*minimisation* or *maximisation*) relevant to the variable being normalised.

3.3.3 Evaluation of the objective functions

Given a solution

$$S = \left\{S1, S2 = \left\{S2_g, for\ each\ scheduled\ group\ g\right\}\right\}$$

We have that:

- Sub-solution $S1$ is composed by a set of nodes on the Layer 1 construction graph;
- Sub-solutions $S2_g$ are composed by sets of edges on the relevant Layer 2 construction graphs.

Each objective functions is therefore computed as follows:

$$TTT(S) = \sum_{c_{L1}^{g} \in S1 \cap C_{L1}} \sum_{e(u,v) \in S2_g} w_{(u,v),TTT} \quad \text{Eq. 7}$$

$$EC(S) = \sum_{c_{L1}^{g} \in S1 \cap C_{L1}} \sum_{e(u,v) \in S2_g} w_{(u,v),EC} \quad \text{Eq. 8}$$

$$CFL(S) = \sum_{c_{L1}^{g} \in S1 \cap C_{L1}} \sum_{e(u,v) \in S2_g} w_{(u,v),CFL}(\Phi_g) \quad \text{Eq. 9}$$

$$TTT(S) = \sum_{c_{L1}^{g} \in S1 \cap C_{L1}} \sum_{e(u,v) \in S2_g} w_{(u,v),ST}(\Phi_g) \quad \text{Eq. 10}$$

$$NTR(S) = \sum_{c_{L1}^{g} \in S1 \cap C_{L1}} w_{s_{L1}^{g}} \quad \text{Eq. 11}$$

The reader is referred to the Sections 3.5 and 3.5.2 for an extensive description of the used notation. We anticipate here that:

- $w_{(u,v),TTT}$ and $w_{(u,v),EC}$ are edge weights of the Layer 2 TEGs, relevant to the travel time and energy consumption respectively, which are statically calculated *a-priori*;
- $w_{(u,v),CFL}(\Phi_g)$ and $w_{(u,v),ST}(\Phi_g)$ are edge weights of the Layer 2 TEGs, relevant to the conflicts number and time stability (minimum buffer time) respectively. They are dynamically calculated depending on the set of Layer 2 sub-solutions $\Phi_g = \{S2_{g'} : S2_{g'} \in S2 \ \wedge g' \prec g\}$, relevant to the groups g' scheduled before g;
- $w_{s_{L1}^{g}}$ are node weights on the Layer 1 graph, relevant to the number of trains belonging to each group, multiplied by the group's priority factor. In case of during Layer 1 exploration a discard-group node is chosen, the relevant weight is equal to 0.

3.3.4 Pheromone update

The multi-objective approach leads to the implementation of a Paretian elitist strategy. The record of the best performing ants so far is implemented by the Pareto Optimal Set of the solution generated by those ants. This set is updated with the solutions generated by ants at each iteration, keeping in memory the non-dominated solutions only. Pheromone-updating solutions are then chosen within the POS according to the mechanism described in Section 3.8.

3.4 Algorithm structure

Pseudocode 1 describes the overall architecture of our MOACO algorithm for the TTP. This algorithm actually returns a Pareto Optimal Set (POS) of non-dominated solutions according to the given set of objective functions.
The POS is firstly initialized to the empty set. Then iterations take place until a termination condition is met (depending on a maximum iterations number or a maximum elapsed time). In each iteration, each colony $\mathbb{C}_o$ firstly updates its set of aggregation weights Λ_o^{iter}. Then each ant of each colony starts its exploration of the construction graphs independently from the other ants. For this reason, the ants' exploration can be easily parallelized.
Once all ants have finished their exploration, each of them having built a solution, a Local Search procedure is applied to these solutions. Local Search performs a further, quick refinement of the iteration's solutions. Finally, the POS is updated with the iteration's solutions and the pheromone is updated in turn by a proper subset of the solutions stored in the POS.

PROCEDURE multiObjectiveAntTTP

```
POS = empty set
iter = 1
elapsedTime = 0
while iIter <= maxIter and elapsedTime <= maxTime:
       iterSolutions = empty set
       for each colony 𝕮_o:
              update Λ_o^iter
              for each ant in 𝕮_o:
                     S{S1, S2}_ant = ant_exploreLayer1()
                     add S{S1, S2}_ant to iterSolutions
```

```
        applyLocalSearch(iterSolutions)
        update POS with iterSolutions
        update pheromone with POS

        iter++
        update elapsedTime
return POS
```

Pseudocode 1. Main architecture of the algorithm.

As anticipated, the exploration of the construction graph by each ant is articulated into two layers. A node on the Layer 1 construction graph is selected, and depending on that node a certain Layer 2 graph (i.e. a TEG) is explored immediately after. Following sections describe this process in details. Table 1 provides a summary of the MOACO parameters.

$Nant$	Number of ants per colony	$\tau_{MIN,L2}$	Lower pheromone bound in L2
$maxIter$	Maximum number of iterations	$\tau_{MAX,L2}$	Upper pheromone bound in 2
$maxTime$	Maximum computation time	$N^{weights}$	N° of main objective aggregation weights per colony
α_{L1}	Pheromone information weight in L1	$\underline{\lambda}$	Min. value of the aggregation weight of the colony's main objective
β_{L1}	Heuristic information weight in L1	n_{SMART}	Smart mode usage period
ρ_{L1}	Pheromone evaporation factor in L1	n_{FAST}	Fast mode usage period
$\tau_{MIN,L1}$	Lower pheromone bound in L1	N_{upd}	Number of updating solutions per region
$\tau_{MAX,L1}$	Upper pheromone bound in L1	$pcSol_{LS}$	Percentage of solution components refined by Local Search
α_{L2}	Pheromone information weight in Layer 2	n_{LS}	Local search usage period
β_{L2}	Heuristic information weight in L2	n_{noLS}	Local search not-usage period
ρ_{L2}	Pheromone evaporation factor in L2		

Table 1. Summary of the MOACO algorithm's parameters.

3.5 Layer 1 exploration

3.5.1 Clique construction

The exploration of Layer 1 defines the sub-solution $S1$, i.e., the sequence in which the groups are scheduled as well as, for optional groups only, whether the group is scheduled or not. We define a directed construction graph $G_{L1} = (N_{L1}, E_{L1})$ whose set of nodes $N_{L1} = \{b_{L1}\} \cup C_{L1} \cup D_{L1}$ is composed by:

- A fictious begin node b_{L1};
- A set C_{L1} composed by the *schedule nodes* c_{L1}^{g} defined for each movable or optional group g;
- A set D_{L1} composed by the *discard nodes* d_{L1}^{g} defined for each optional group g.

We say that a node d_{L1}^{g} is the *dual* of the node c_{L1}^{g}, if it is relevant to the same optional group g, and vice-versa. We also define two sets of nodes, M_{L1} and O_{L1}, such that

- $N_{L1} = \{b_{L1}\} \cup M_{L1} \cup O_{L1}$;
- M_{L1} contains all the nodes relevant to the scheduling of a movable group;
- O_{L1} contains all the nodes relevant to the scheduling or the discarding of an optional group.

Directed edges are defined between:

- b_{L1} and all the other nodes;
- All the sorted pairs of *schedule* or *discard* nodes, with the exception of those pairs of nodes relevant to the same optional group.

A clique in G_{L1} is a Layer 1 solution. Figure 2 shows an example of graph and solution. Here, four train groups are given as input. Group 2 is optional. The bold clique on the graph shows that Group 4 is scheduled first, before Group 1, Group 2, which is not actually scheduled, and Group 3.

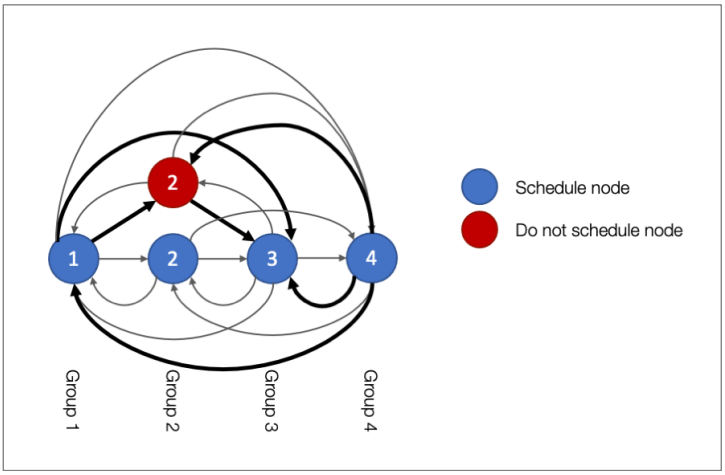

Figure 2. Example of a Layer 1 graph with 3 movable and 1 optional train groups. Bold edges represent a solution-clique given by the sequence {S4, S1, NS2, S3}. The virtual begin node and the outgoing edges are omitted for readability.

Pseudocode 2 implements the exploration of G_{L1}. The begin node b_{L1} is firstly inserted into the list $S1$, containing the partial solution on Layer 1. The candidates set is initialized with all the nodes in $N_{L1} - \{b_{L1}\}$. Then the Overlap Index (see next Section 3.5.2) is computed for each edge $e(u,v) \in E_{L1}, u \in S1, v \in CAND$. A candidate is now chosen according to the pseudo-random criterion, it is appended to $S1$ and removed from the candidates set. If its dual node exists, it is removed from the candidates set as well. Finally, if the chosen candidate belongs to C_{L1}(it's a *schedule node*), the procedure ***ant_scheduleGroup*** is invoked, which performs the actual scheduling of that group in Layer 2.

```
PROCEDURE ant_exploreLayer1()
S1 = [bL1]
S2 = empty list
candidates = NL1 - {bL1}
While candidates is not empty:
        For each v in candidates:
                Compute oiv(S1)
        ug_L1 = randomly choose an item from candidates
        remove ug_L1 from candidates
        if dual(ug_L1) exists, remove it from candidates
        append ug_L1 to S1
        If CL1 contains ug_L1: # ug_L1 is a scheduled node
                S2g = ant_scheduleGroup(S2, g)
                append S2g to S2
```

Pseudocode 2. Exploration of the Layer 1 graph.

Each node $v \in N_{L1} - \{b_{L1}\}$, relevant to the group g, is qualified by the heuristic information for the objective *NTR*, defined as a node weight:

$$w_{NTR,v} = \begin{cases} ntr_g \cdot pr_g \; if \; v \in C_{L1} \\ 0 \; otherwise \end{cases} \quad \text{Eq. 12}$$

At each choosing step, this is normalised according to Eq. 4,

$$\bar{w}_{NTR,v} = norm\left(w_{NTR,v}, best_{NTR}, worst_{NTR}\right) \quad \text{Eq. 13}$$

A second type of heuristic information is used during the Layer 1 exploration. Each time a new candidate node is to be chosen, each edge $e(u,v) \in E_{L1}, u \in S1, v \in CAND$, connecting a node already in the partial solution and one still to be chosen, is qualified by a heuristic information named *Overlap Index* $oi_{(u,v)}$. OI estimates how much a group, still to be scheduled, would potentially be in conflict with an already scheduled one, being a conflict a simultaneous utilisation (or two utilisations

not separated by a proper minimum headway) of the same infrastructure resource (line stretch, station track). In case of $u = b_{L1}$ or $\{u, v\} \cap D_{L1} \neq \emptyset$, $oi_{(u,v)}$ is equal to 0. Otherwise, $oi_{(u,v)}$ is calculated according to the procedure described in Appendix 1.
For each candidate v, we the define the value oi_v as follows

$$oi_v = oi_v(S1) = \sum_{u \in S1} oi_{(u,v)} \qquad \text{Eq. 14}$$

We assume that OI acts as a proxy for the objectives TTT, EC, ST, CFL. Two groups with high OI would likely originate residual conflicts, or, in case they can be avoided, their courses would be scheduled in such a way that:

- The stability would be likely affected;
- The total travel time would increase in order to avoid conflicts (e.g., due to station crossings);
- The energy consumption would increase for the same reason (e.g. because of extra station stops for crossings).

As a consequence of that, we assume that

$$w_{TTT,v} = w_{EC,v} = w_{ST,v} = w_{CFL,v} = oi_v(S1) \qquad \text{Eq. 15}$$

Where $S1$ is the Layer 1 sub-solution built so far.
The overlap index is then normalised by applying the following equation

$$\overline{w}_{TTT,v} = \overline{w}_{EC,v} = \overline{w}_{ST,v} = \overline{w}_{CFL,v} = \overline{oi}_v(S1) = \sum_{u \in S1} norm(oi_{(u,v)}, best_{oi}, worst_{oi}) \qquad \text{Eq. 16}$$

We first normalise each $oi_{(u,v)}$ and then sum them for consistency with the way in which pheromone contribution is calculate, remembering that the normalisation operator returns a value in $[\tau_{MIN}, \tau_{MAX}]$. Following the clique pheromonal strategy, the pheromone factor of each candidate $v \in CAND$ is the sum of the pheromone laying on the edges connecting the candidate to all the already chosen nodes,

$$\tau_v = \tau_v(S1) = \sum_{u \in S1} \tau_{(u,v)}$$

The aggregated heuristic factor finally used is

$$\overline{\eta}_v = \overline{\eta}_v(S1) = \\ = \lambda_{NTR} \cdot \overline{w}_{NTR,v} + \sum_{o \in O - \{NTR\}} \lambda_o \cdot \overline{w}_{o,v}(S1) = \lambda_{NTR} \cdot \overline{w}_{NTR,v} + (1 - \lambda_{NTR}) \cdot \overline{oi}_v(S1) \qquad \text{Eq. 17}$$

with $\lambda_o \in \Lambda^{iter}$.
Normalisation in Eq. 17 requires to provide the best and worst values. They are calculated as follows

$$best_{oi} = \begin{cases} MAX\left(oi_{(u,v)}(S1), \forall u \in S1, \forall v \in CAND \cap M_{L1}\right) \; if \; v \in M_{L1} \\ MIN\left(oi_{(u,v)}(S1), \forall u \in S1, \forall v \in CAND \cap O_{L1}\right) \; if \; v \in O_{L1} \end{cases}$$

$$worst_{oi} = \begin{cases} MIN\left(oi_{(u,v)}(S1), \forall u \in S1, \forall v \in CAND \cap M_{L1}\right) \; if \; v \in M_{L1} \\ MAX\left(oi_{(u,v)}(S1), \forall u \in S1, \forall v \in CAND \cap O_{L1}\right) \; if \; v \in O_{L1} \end{cases}$$

This definition of the heuristic factor produces the following effects. On one hand, in case of movable train groups, high-priority, highly-populated and highly-conflictive groups are favoured to be scheduled before other ones. Since a movable train group must be scheduled in any case, if it is already significantly conflictive, to schedule it before less conflictive groups would contribute to reduce residual conflicts. Differently, if it was scheduled lastly, residual conflicts are likely to increase.
On the other hand, in case of optional train groups, a balance is implemented between the benefits of not to schedule the group (reduction of residual conflicts) and those of scheduling it (increase of the number of scheduled trains). To this purpose, lowly-conflictive "schedule" candidates are

favoured over highly-conflictive "schedule" candidates. During next choosing steps, an already highly-conflictive candidates would only increase its OI, thus increasing its probabilities to be discarded as well.
Finally, at each construction step, the next node is randomly chosen from the candidates list with a probability

$$p_v = \frac{{\tau_v}^{\alpha_{L1}} \cdot {\overline{\eta}_v}^{\beta_{L1}}}{\sum_{v' \in CAND} \left({\tau_{v'}}^{\alpha_{L1}} \cdot {\overline{\eta}_{v'}}^{\beta_{L1}} \right)} \quad \text{Eq. 18}$$

3.5.2 Computation of the Overlap Index

The overlap index estimates how much a yet to be scheduled train group would be conflictive with already scheduled ones.
Given a set $\bar{S}$ of already scheduled groups and a group g to be scheduled, we denote as W the set of already scheduled trains (belonging to groups in $\bar{S}$) and C the set of trains belonging to group g. Given two trains $w \in W$ and $c \in C$, we consider the infrastructure resources shared by the two, being a resource a line section or a station track. On each of them ($\mathfrak{r}$), we calculate the conflict probability $po_{\mathfrak{r},c,w}$ as the probability of having an overlap between the two "utilisation blocks" of w and c. An "utilisation block" represents the time a resource is utilised by a single train, and cannot be utilised by other ones for safety reasons. On the one hand, the utilisation block of w is fixed, being w already scheduled. It is defined by a start time $u_{start,w}$ and an end time $u_{end,w}$. On the other hand, the start time of c's utilisation block can vary within a set of feasible times $\left[\underline{u_{start,c}}, \overline{u_{start,c}}\right]$ depending on the train journey, stops and running time capabilities. Given this start time, the separation between utilisations of w and c can be computed. If it is smaller than the minimum headway time, computed accounting for travel directions, then a conflict occurs. If w uses $\mathfrak{r}$ before c, the minimum headway time is a constant, as it is a function of the first train travel and everything is already fixed for w. Otherwise, it depends on the speed and the EEE chosen for c and it can vary in $\left[\underline{h_c}, \overline{h_c}\right]$. Let $\Delta_h = \overline{h_c} - \underline{h_c}$ and $\Delta_{start} = \overline{u_{start,c}} - \underline{u_{start,c}}$ be the duration of the two relevant time intervals. We assume that c utilisation start time $\underline{u_{start,c}}$ and minimum headway if c precedes w h_c are two random variables with independent uniform distributions in the intervals $\left[\underline{u_{start,c}}, \overline{u_{start,c}}\right]$ and $\left[\underline{h_c}, \overline{h_c}\right]$ respectively.
The probability that the two blocks overlap is given by the probability that the combination of the choices of c utilisation start time and minimum headway if c uses r first overlaps the utilisation by w and the minimum headway if the latter goes first. $po_{\mathfrak{r},c,w}$ is equal to 0 if $\overline{u_{s,c}} \leq u_{s,w}$ or $u_{e,w} \leq \underline{u_{s,c}}$, otherwise we formalise it as

$$\int_A^B \frac{1}{\Delta_{start}} \cdot \left(\int_{u_{s,w}}^{u_{s,c}+\overline{h_c}} \frac{1}{\Delta_h} dh_c \right) du_{s,c} + \int_B^C \frac{1}{\Delta_{start}} du_{s,c} \quad \text{Eq. 19}$$

With

$$A = \max\left(\underline{u_{s,c}}, u_{s,w} - \overline{h_c}\right) \quad \text{Eq. 20}$$

$$B = \min\left(\overline{u_{s,c}}, u_{s,w} - \underline{h_c}\right) \quad \text{Eq. 21}$$

$$C = \min\left(\overline{u_{s,c}}, u_{e,w}\right) \quad \text{Eq. 22}$$

Eq. 19 can be reduced to:

$$po_{\mathfrak{r},c,w} = \frac{\overline{u_{size,c}} - u_{start,w}}{\Delta_{size}\Delta_{start}}(B - A) + \frac{B^2 - A^2}{2\Delta_{size}\Delta_{start}} + \frac{C - B}{\Delta_{start}} \quad \text{Eq. 23}$$

The overlap index $oi_{c,w}$ of a train to be scheduled c with regards to an already scheduled one w is defined as

$$oi_{c,w} = \sum_{l \in CLS} po_{l,c,w} + \sum_{s \in CST} \left(\sum_{\tau \in CTR_s} po_{\tau,c,w} \right) \frac{1}{\left| \mathrm{T}_{g,l} \right|} \quad \text{Eq. 24}$$

Where CLS and CST are the set of the line section and stations used by both c and w ,CST is the set of stations used by both c and w , respectively. CTR_s, for a station $s \in CST$, is the set of tracks used by both c and w. This has always cardinality equal to 1, since the track for w is already defined. Finally, $\left| \mathrm{T}_{g,l} \right|$ is the cardinality of the set of available tracks in station s for course c.
The overlap index $oi_{g,S}$ of a group g with respect to a set of already scheduled groups $\bar{S}$ is computed as the sum of the overlap index for each pair of trains.

3.6 Layer 2 exploration

3.6.1 Definition of the Time Expanded Graph

The exploration in Layer 2 produces the actual scheduling of a given train group. It is performed by choosing a path on a Time Expanded (directed) Graph (TEG). TEG's nodes represent *discrete timetable events* for the first train of the group, i.e. the arrival (or departure) at (from) a certain station, at a certain time, at a certain station track, with or without a stop. The definition of discrete timetable events requires to discretize time. TEG's directed edges represent transitions, in stations or line stretches. A station transition models a train stopping or passing at a certain station, while a line transition models a train travelling from a station to the following one. A path on the TEG completely describes the timetable of the first train of each group. During Layer 2 exploration, a path pheromonal strategy is implemented. Having assumed that trains of the same group are strictly periodic, the timetable of the other trains is simply obtained by shifting in time of multiple of the period the timetable of the first one.
For group g, the TEG G_{L2}^{g} is defined by a set of nodes N_{L2}^{g} and a set of edges E_{L2}^{g}

$$G_{L2}^{g} = \left(N_{L2}^{g}, E_{L2}^{g} \right) \quad \text{Eq. 25}$$

For readability, in the following we omit the pedix "L2". The set of nodes is defined as follows

$$N_g = \sum_{l \in J_g} \left(A_{g,l} + D_{g,l} \right) \quad \text{Eq. 26}$$

For each location l of the group's journey J_g, $A_{g,l}$ and $D_{g,l}$ are the arrival and departure node sets respectively. The elements of these sets are defined considering the lower and upper bounds for the arrival/departure event, the time discretization and the available station tracks and pass/stop events in that location. Notice that we have two exceptions to the following definitions for the first and last locations of the journey, $\underline{l}$ and $\overline{l}$ respectively.

$$A_{g,l} = \sum_{\gamma \in \Gamma_{g,l}} \sum_{\tau \in \mathrm{T}_{g,l}} \sum_{t \epsilon DTARR_{g,l}} a_{g,l,\gamma,\tau,t} \quad \forall\, l \in J_g - \{\overline{l}\} \quad \text{Eq. 27}$$

$$D_{g,l} = \sum_{\gamma \in \Gamma_{g,l}} \sum_{\tau \in \mathrm{T}_{g,l}} \sum_{t \epsilon DTDEP_{g,l}} d_{g,l,\gamma,\tau,t} \quad \forall\, l \in J_g - \{\underline{l}\} \quad \text{Eq. 28}$$

Where $a_{g,l,\tau,t}$ ($d_{g,l,\tau,t}$) represents the discrete arrival (departure) event node at (from) with pass/stop mode γ at track τ in location l at time t, where $\Gamma_{g,l}$ is the set of available pass/stop modes at location l and $\mathrm{T}_{g,l}$ the set of available station tracks.
The sets $DTARR_{g,l}$ and $DTDEP_{g,l}$ contain the available discretized times for the arrival and departure events:

$$DTARR_{g,l} = \left\{ t \in \mathbb{Z} : t \bmod \varphi_g = 0 \ \wedge\ t \geq \underline{arr}_{g,l} \ \wedge\ t \leq \overline{arr}_{g,l} \right\} \quad \text{Eq. 29}$$

$$DTDEP_{g,l} = \left\{ t \in \mathbb{Z} : t \bmod \varphi_g = 0 \ \wedge \ t \geq \underline{dep}_{g,l} \ \wedge \ t \leq \overline{dep}_{g,l} \right\} \quad \text{Eq. 30}$$

Where:

- φ_g is the time discretization for group g;
- $\underline{arr}_{g,l}$ and $\overline{arr}_{g,l}$ are the lower and upper bounds respectively for the arrival at l;
- $\underline{dep}_{g,l}$ and $\overline{dep}_{g,l}$ are the lower and upper bounds respectively for the departure from l;

In the first and last locations of the journey these sets are defined as follows

$$D_{g,\underline{l}} = \{b_g\} \quad \text{Eq. 31}$$

$$A_{g,\overline{l}} = \{f_g\} \quad \text{Eq. 32}$$

Where b_g and f_g are two fictious *begin* and *end* nodes respectively.
Directed edges in $e(u,v) \in E_{L2}^g$ represent timetable transitions linking consecutive discrete timetable events. For each departure node from the first location $\underline{l}$ $d_{g,\underline{l},\gamma,\tau,t} \in D_{g,\underline{l}}$, we define an edge $e(b_g, d_{g,\underline{l},\gamma,\tau,t})$ connecting the fictious begin node to it. Similarly, for each arrival node at the last location $\overline{l}$ $a_{g,\overline{l},\gamma,\tau,t} \in A_{g,\overline{l}}$, we define an edge $e(a_{g,\overline{l},\gamma,\tau,t}, f_g)$ connecting it to the fictious end node.
In each location $l \neq \underline{l}$, between the pair of nodes $a_{g,l,\gamma,\tau,t}$ and $d_{g,l,\gamma',\tau',t'}$ it exists an edge if and only if all the following conditions are simultaneously true:

- $\gamma = \gamma'$;
- $\tau = \tau'$;
- $(t' - t) \geq \underline{stop}_{g,l}$ and $(t' - t) \leq \overline{stop}_{g,l}$, being $\underline{stop}_{g,l}$ and $\overline{stop}_{g,l}$ the lower and upper bounds for the stop time respectively.

Between two departure and arrival nodes $d_{g,l,\gamma,\tau,t}$ and $a_{g,l',\gamma',\tau',t'}$, relevant to consecutive locations l and l', l preceding l' in J_g by means of the infrastructure edge e in direction dir , it exists a TEG edge if and only if all the following conditions are met:

- $(t' - t) \geq \underline{rt}_{g,e}$;
- $(t' - t) \geq \overline{rt}_{g,e}$;
- $(t' - t) \geq mrt_{e,dir,\varepsilon,\tau,\tau'}^{rs_{g,l}}$

Where $\underline{rt}_{g,e}$ and $\overline{rt}_{g,e}$ are the minimum and maximum run times admitted for the group's trains on the infrastructure edge e, and $mrt_{e,dir,\varepsilon,\tau,\tau'}^{rs_{g,l}}$ is the minimum technical run time for the rolling stock $rs_{g,l}$ with the considered combination of station tracks and the EEE $\varepsilon = EEE(\gamma,\gamma')$[1].

$$mrt_{e,dir,\varepsilon,\tau,\tau'}^{rs_{g,l}} = mrt_{e,dir,\varepsilon}^{rs_{g,l}} + \Delta rt_{rs_{g,l}}^{\tau} + \Delta rt_{rs_{g,l}}^{\tau'} \quad \text{Eq. 33}$$

Figure 3 provides a graphical example of the TEG relevant to a group travelling through 3 stations. The fictious begin and end nodes are highlighted, as well as the arrival and departure nodes in each location. While in stations 1 and 3 just one track can be used, in station 2 two tracks are available. In all the three stations on the $STOP$ station mode shall be used, and in station 2 the minimum stop time is greater than 0.

[1] $EEE(\gamma,\gamma')$ is the operator that, given two pass/stop station modes in consecutive locations γ,γ', returns the consistent Edge Extremity Event ε. E.g. $\gamma = PASS,\ \gamma' = STOP \implies \varepsilon = PASS/STOP$

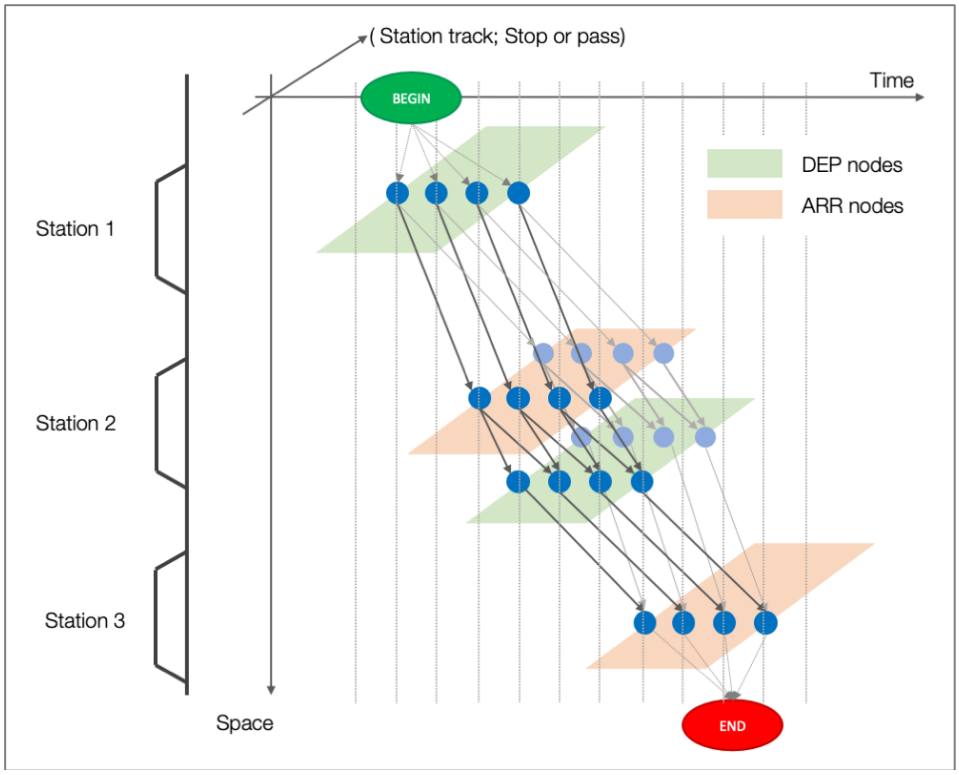

Figure 3. Graphical example of a Time Expanded Graph.

3.6.2 Weights on the TEG

Each edge $e(u, v)$ is characterised by the following set of weights $w_{(u,v),o}$, one for each objective $o \in O - \{NTR\}$:

- $w_{(u,v),TTT}$ is the duration of the transition represented by that edge, multiplied by the number of trains ntr_g belonging to group g.
- $w_{(u,v),EC}$ is the energy consumed by the group's trains performing that transition, multiplied by the number of trains ntr_g belonging to group g. In case of station transitions (nodes u, v referring to the same location), $w_{(u,v),EC}$ is defined equal to 0;
- $w_{(u,v),CFL}$ is defined as the number of additional conflicts with already scheduled train groups that would arise in case of the group's trains would perform that transition.
- $w_{(u,v),ST}$ is defined as the minimum buffer time with already scheduled train groups in case of the group's trains performing that transition.

$w_{(u,v),TTT}$ and $w_{(u,v),EC}$ are calculated a-priori during the TEG's construction. They are a static attribute of the TEG's edges. $w_{(u,v),TTT}$ is the difference between the times associated to nodes v and u, i.e. the run time (line transition) or the stop time (station transition) multiplied by the number of trains belonging to the group. Only in case of line transitions, $w_{(u,v),EC}$ is the energy consumed by the rolling stock used by the group when travelling from the pairs of locations associated to nodes u and v with the above-mentioned run time, multiplied by the number of trains belonging to the group.

$$w_{(u,v),CFL} = w_{(u,v),CFL}(\Phi_g) \quad \text{Eq. 34}$$

$$w_{(u,v),ST} = w_{(u,v),ST}(\Phi_g) \quad \text{Eq. 35}$$

$w_{(u,v),CFL}$ and $w_{(u,v),ST}$ are dynamically calculated each time a TEG is to be explored by an ant. They depend on the set Φ_g of partial solutions $S2_g$, found by the same ant for already scheduled train

groups g'. In particular, if $e(u,v)$ relies on a certain infrastructure resource $\mathfrak{r}$ (being $\mathfrak{r}$ a line stretch or a station track), a comparison is performed between e and each $e' \in S2_{g'}$, $S2_{g'} \in \Phi_g$, such that e' relies on $\mathfrak{r}$ too. This comparison calculates the number of conflicts and the minimum buffer time between the trains of g' (travelling according to their already scheduled timetable) and the trains of g in case the latter would use transition $e(u,v)$ on $\mathfrak{r}$.
It is worthwhile to highlight that, in principle, this information could be calculated *a priori*, comparing each possible couple of edges belonging to different TEGs but relying on the same resource. However, experiments revealed that the amount of memory needed to store this information exceeds acceptable levels with real-size instances. Hence the decision to calculate it dynamically, accepting higher computation times per iteration.

3.6.3 TEG pruning

The such constructed TEG undertakes a "pruning" process in which all the "leaf" nodes, different from the fictious end node f_g, are removed. Leaf nodes are nodes with no outgoing edges. The pruning procedure is described in Pseudocode 3. After pruning, for each remaining node of the TEG, at least one path exists linking the fictious start and end nodes and containing the node in point.

```
PROCEDURE pruneTheTEG(TEG)
N = nodes in TEG different from b_g and f_g
S = nodes in N with no outgoing edges
while S is not empty:
        remove from TEG the nodes in S with all their ingoing edges
        N = N – S
        S = nodes in N with no outgoing edges
```

Pseudocode 3. TEG pruning procedure.

A pruned TEG occupies less memory and is required for the implementation of the FAST exploration, since it ensure that the fictious end node is reachable from all the TEG's node.

3.6.4 FAST exploration

The TEG can be explored in a so-called *fast mode*, formalized by Pseudocode 4. Starting from the begin node, a path is constructed, choosing the next node within the neighbours of the last chosen one, until the end node is reached.
In fast mode, a simple mechanism is implemented to push the exploration towards conflict-free scheduling. At each step, the $noCflCands$ set is defined as that containing all the neighbours of the last-chosen node which can be reached via conflict-free transitions. If $noCflCands$ is empty, the candidates set $CAND$ is defined as the entire neighbourhood of the last-chosen node. Otherwise, $CAND = noCflCands$. Then, the next node is chosen within the $CAND$ set. Notice that this mechanism does not ensure that, if one conflict-free path exists, it is returned by the exploration.

```
PROCEDURE ant_scheduleGroupFast(S2, g)
S2_g = [f_g]
u = f_g
while u is not b_g:
        for each v such that u is in neighbours of v:
                compute w_(v,u),CFL and w_(v,u),ST
        noCflCands = {v : u in neighbours of v such that w_(v,u),CFL=0}
        if noCflCands is not empty:
                cand = noCflCands
        else:
                cand = {v : u in neighbours of v}
        u = randomly choose an item from candidates
        push u into S2_g
return S2_g
```

Pseudocode 4. Exploration of a TEG in FAST mode.

Each node $v \in CAND$ in the candidates set is randomly chosen with probability

$$p_v = \frac{{\tau_{(u,v)}}^{\alpha} \cdot {\overline{\eta}_{(u,v)}}^{\beta}}{\sum_{v' \in CAND} \left({\tau_{(u,v')}}^{\alpha} \cdot {\overline{\eta}_{(u,v')}}^{\beta} \right)} \qquad \text{Eq. 36}$$

Where $\tau_{(u,v)}$ is the pheromone present on the edge connecting u to v, and $\overline{\eta}_{(u,v)}$ is the aggregated normalised heuristic factor

$$\overline{\eta}_{(u,v)} = \overline{\eta}_{(u,v)}(\Phi_g) = \\ = \lambda_{TTT} \cdot \overline{w}_{(u,v),TTT} + \lambda_{EC} \cdot \overline{w}_{(u,v),EC} + \lambda_{CFL} \cdot \overline{w}_{(u,v),CFL}(\Phi_g) + \lambda_{ST} \cdot \overline{w}_{(u,v),ST}(\Phi_g) \qquad \text{Eq. 37}$$

Where:

- $\overline{w}_{(u,v),TTT} = norm(w_{(u,v),TTT}, best_{TTT}, worst_{TTT})$, being $best_{TTT} = \min(w_{(u,v'),TTT}, v' \in CAND)$, $worst_{TTT} = \max(w_{(u,v'),TTT}, v' \in CAND)$;
- $\overline{w}_{(u,v),EC} = norm(w_{(u,v),EC}, best_{EC}, worst_{EC})$, being $best_{EC} = \min(w_{(u,v'),EC}, v' \in CAND)$, $worst_{EC} = \max(w_{(u,v'),EC}, v' \in CAND)$;
- $\overline{w}_{(u,v),CFL}(S2) = norm(w_{(u,v),CFL}(S2), best_{CFL}, worst_{CFL})$, being $best_{CFL} = \min(w_{(u,v'),CFL}(S2), v' \in CAND)$, $worst_{CFL} = \max(w_{(u,v'),CFL}(S2), v' \in CAND)$;
- $\overline{w}_{(u,v),ST}(S2) = norm(w_{(u,v),ST}(S2), best_{ST}, worst_{ST})$, being $best_{ST} = \max(w_{(u,v'),ST}(S2), v' \in CAND)$, $worst_{ST} = \min(w_{(u,v'),ST}(S2), v' \in CAND)$.

3.6.5 SMART exploration

Pseudocode 5 formalizes the so-called *smart exploration mode* of the Layer 2 TEG. This mode exploits a more complete heuristic information, which allows to restrict the exploration to those paths which minimise (and possibly avoid) the occurrence of conflicts. With this approach, the TEG's nodes are firstly labelled by means of a modified Dijkstra algorithm, and then a path is computed making use of these labels. Pseudocode 5 contains the overall framework of this approach.

```
PROCEDURE ant_scheduleGroupSmart(S2, g)
labelledTEG = labelNodes(S2, g)
S2_g = getTheRandomPath(labelledTEG)
return S2_g
```

Pseudocode 5. Exploration of a TEG in SMART mode.

The idea beneath the labelling procedure is to firstly activate just those TEG edges (and nodes) which belongs to paths featuring the minimum possible number of conflicts with already scheduled train groups. This information is provided by the label *conflict distance*, which is the minimum cost of a path linking each node to the begin one, computed utilising as edge costs the conflict weights $w_{(u,v),CFL}$. As already explained, the latter is the number of additional conflicts with already scheduled groups that the utilisation of a certain edge implies. Secondly, each of these nodes is labelled with its normalised aggregated distance from the begin node. This distance is the minimum cost of a path linking each node to the begin node, computed utilising as edge costs the normalised aggregated heuristic factors $\overline{\eta}_{(u,v)}$ calculated as in Section 3.6.4.

Pseudocode 6 formalizes the labelling procedure. Each node is initialized setting to positive infinity the labels *conflict distance* (`cflDist`) and *aggregated normalised distance* (`aggNormDist`). The label *previous* is introduced too, as it will be used by the Local Search (see Section 3.7). Then the label *conflict distance* is computed for each node of the TEG (Part 1). The conflict distance of the end node f_g is therefore the minimum possible number of additional conflicts that the scheduling of the current group would cause with regard to the already scheduled train groups. If it is equal to 0, there exists at least one path in the TEG that does not introduce any additional conflict. Smart exploration mode therefore restricts any further search to those minimum-conflicts paths only.

To this purpose, we define the set *active nodes* containing all the nodes featuring a conflict distance minor or equal to the conflict distance of the end node. Then, the set *active edges* contains the TEG's edges $e(u, v)$ such that u and v are both contained in *active nodes* and that the conflict distance of v is greater or equal than the conflict distance of u.

In Part 2 each node of the subgraph composed by *active nodes* and *active edges* is labelled with its aggregated normalised distance. This will actually be used as the heuristic information during the random construction of the path linking b_g to f_g.

```
PROCEDURE labelNodes(S2, g)
# INITIALIZATION
for each u in Ng - {b_g}
        cflDist(u) = INF
        aggNormDist(u) = INF
        previous(u) = None

# PART1: activation of the minimum conflict paths edges
nodesToLabel = Ng
cflDist(b_g) = 0
while nodesToLabel is not empty:
        u = remove from nodesToLabel the node with minimum cflDist(u)
        for v in neighbours(u):
                alt = cflDist(u) + w_(u,v),CFL
                if alt < cflDist(v):
                        cflDist(v) = alt

activeNodes = {u in Ng : cflDist(u) <= cflDist(f_g)}
activeEdges = {e(u, v) in Eg : u,v in activeNodes and cflDist(u) <= cflDist(v)}

# PART2: labelling with the aggregated normalised distance
nodesToLabel = activeNodes
aggNormDist(b_g) = 0
while nodesToLabel is not empty:
        u = remove from nodesToLabel the node with minimum aggNormDist(u)
        for v in neighbours(u) ∩ activeEdges
                alt = aggNormDist(u) + η̄_(u,v)
                if alt < aggNormDist(v):
                        aggNormDist(v) = alt
                        previous(v) = u
return the labelled TEG
```

Pseudocode 6. Labelling of the TEGs nodes in SMART mode.

```
PROCEDURE getTheRandomPath(labelledTEG)
S2_g = [f_g]
u = f_g
while u is not b_g:
        u = randomly choose an item from {v : u in neighbours(v) : e(v,u) in
        activeEdges}
        push u into S2_g
return S2_g
```

Pseudocode 7. Computation of the $S2_g$ partial solution in SMART mode.

In Pseudocode 7 the choice of the next candidate is restricted to those neighbours of last-chosen node u which are linked to u by and edge in the *active edges set.* Each of these nodes v has a probability to be chosen equal to

$$p_v = \frac{{\tau_{(u,v)}}^{\alpha} \cdot \overline{aggNormDist}(v)^{\beta}}{\sum_{v' \in CAND} \left({\tau_{(u,v')}}^{\alpha} \cdot \overline{aggNormDist}(v')^{\beta} \right)} \qquad \text{Eq. 38}$$

Where $\tau_{(u,v)}$ is the pheromone on the edge (u, v), and $\overline{aggNormDist}(v)$ is defined as

$$\overline{aggNormDist}(v) = \frac{aggNormDist(v)}{GD(b_g, v) - 1} \qquad \text{Eq. 39}$$

given that the geodesic distance $GD(u, u')$ between two nodes of a graph is the minimum number of edges in a shortest path connecting them. We use $\overline{aggNormDist}(v)$ instead of

$aggNormDist(v)$ in order to improve the comparability of the values of pheromonal and heuristic information. The aggregated normalised distance of a node v is the sum of the aggregated normalised costs of the edges of the shortest path $SP(b_g, v)$ (shortest according to the aggregated normalised costs) linking b_g to v .

$$aggNormDist(v) = \sum_{e(u,v) \in SP(b_g,v)} \overline{\eta}_{(u,v)} \qquad \text{Eq. 40}$$

The aggregated normalised cost of each edge $e(u,v)$, $u \neq b_g$ is always $\tau_{MIN} \leq \overline{\eta}_{(u,v)} \leq \tau_{MAX}$ (according to the defined normalisation and aggregation operators), while all the edges departing from b_g have aggregated normalised cost equal to 0. The shortest path contains one and only one edge departing from b_g, therefore the following inequalities are always valid

$$\left(GD\left(b_g, v\right) - 1\right) \cdot \tau_{MIN} \leq \sum_{e(u,v) \in SP(b_g,v)} \overline{\eta}_{(u,v)} \leq \left(GD\left(b_g, v\right) - 1\right) \cdot \tau_{MAX} \qquad \text{Eq. 41}$$

and

$$\tau_{MIN} \leq \overline{aggNormDist}(v) = \frac{\sum_{e(u,v) \in SP(b_g,v)} \overline{\eta}_{(u,v)}}{GD\left(b_g, v\right) - 1} \leq \tau_{MAX} \qquad \text{Eq. 42}$$

3.6.6 Blending of SMART and FAST Layer 2 exploration modes

The two exploration modes both produce the scheduling of the given train group. Their performance (w.r.t. the optimisation of the objective functions) as well as they computation time are likely to be rather different, as it results from the tuning presented in Section 5.2.1. We blend their usage, according to the current iteration index $iter \in [1, maxIter]$ and to the two parameters *smart* and *fast mode utilisation periods,* n_{SMART} and n_{FAST} respectively:

- If the modulus of $iter - 1$ and the sum of n_{SMART} and n_{FAST} is minor than n_{SMART}, the smart mode is used;
- Otherwise, the fast mode is used.

This permits to use the smart mode for n_{SMART} consecutive iterations and the fast mode for the following, n_{FAST} ones. To set to 0 these parameters permits to exclude one mode. If both are set to 0, the smart mode is used by default. Pseudocode 8 formalizes this approach.

```
PROCEDURE ant_scheduleGroup(S2, g)
m = mod((iter -1 ),  nSMART + nFAST)
if m <nSMART:
       S2g = ant_scheduleGroupSmart(S2, g)
else:
       S2g = ant_scheduleGroupFast(S2, g)
return S2g
```

Pseudocode 8. Blending strategy of the smart and fast Layer 2 exploration modes.

3.7 Local search

In this section we introduce a Local Search technique to further improve a solution obtained by ants through the randomized exploration of Layer 1 and Layer 2 graphs.

Let $S = \{S1, S2\}$, $S2 = \left\{S2_g : g \in G \wedge c_{L1}^g \in S1 \cap C_{L1}\right\}$ be the solution we want to improve with Local Search. Solution components in $S1$ provide the order in which groups are scheduled, as well as which groups are scheduled and which groups are discarded. For the scheduled groups, components in $S2$ provide the actual scheduling (timetable), resulting from both the heuristic information (strongly affected by the conflicts with already-scheduled train groups) and the pheromone trail. In principle, we can look at the pheromone trail as a "memorized experience" which would lead the current group towards a scheduling which would help the scheduling process of the following ones.

The idea beneath the proposed Local Search is to firstly define a subset $L \subseteq S1 \cap C_{L1}$. For each component in L we reschedule the relevant group, considering during the TEG exploration not only the groups scheduled before it, but all the scheduled groups in $S1$, with their scheduled timetable. On the one hand, this would allow for a reduction of the number of residual conflicts, which is

actually the main target of the local search. On the other hand, such a reduction would likely affect the other objectives.
Pseudocode 9 formalizes this process. Each component of L is considered, following the ordering given by $S1$. The sub-solution $S2_g$ of that group is removed from the list of the Layer 2 sub-solutions. Then, the labelling procedure of the group g TEG is invoked. Finally, an improved partial solution $S2'_g$ is computed and added to the new set $S2'$ of Layer 2 partial solutions. A new improved solution $S' = \{S1, S2'\}$ is returned at the end of the procedure.

```
PROCEDURE localSearch(S{S1, S2})
define L
S2' = S2
for each c^g_L1 in L:
        remove S2_g from S2'
        labelledTEG = labelNodes(S2', g)
        S2_g' = getTheShortestPath(labelledTEG)
        add S2_g' to S2'
return S' = {S1, S2'}
```

Pseudocode 9. Local Search procedure.

The improved partial solution $S2'_g$ for group g is obtained, as in Pseudocode 10, making use of the labels *previous* computed during the nodes labelling procedure (see Pseudocode 6 in Section 3.6.5). To this purpose, it acts as a Dijkstra algorithm since it returns the path between the end and begin nodes with the minimum aggregated normalised costs, searched between those paths featuring the minimum conflict cost.

```
PROCEDURE getTheShortestPath(labelledTEG)
S2'_g = [f_g]
u = f_g
while u is not b_g:
        u = previous(u)
        push u into S2'_g
return S2'_g
```

Pseudocode 10. Computation of the $S2'_g$ improved partial solution during Local Search.

We govern the utilisation of this Local Search by means of three parameters $pcSol_{LS}$, n_{LS} and n_{noLS}. On one side, $pcSol_{LS}$ indicates, as a percentage of the cardinality of $S1 \cap C_{L1}$, how many solutions of $S1 \cap C_{L1}$ will be refined by the Local Search. L is then populated by the first $\left\lceil |S1 \cap C_{L1}| \cdot \frac{pcSol_{LS}}{100} \right\rceil$ solutions of $S1 \cap C_{L1}$. On the other side, we impose the utilisation of the Local Search for n_{LS} consecutive iterations, followed by n_{noLS} iterations in which Local Search is not applied.

3.8 Pheromone update

At the end of each iteration, the Pareto Optimal Set is firstly updated with the newly obtained solutions. Previously obtained solutions of lower quality, i.e. those dominated by recently obtained ones, are discarded. Finally, POS solutions update colonies' pheromone trails according to an *update-by-region* strategy. This strategy imposes that the POS is firstly split into regions, one for each colony. N_{upd} (with $N_{upd} \geq 1$) solutions of each region will then update the corresponding colony.
To this purpose, the POS is split into equal-cardinality subsets R_o (regions), in such a way that solutions in each subset are the best according to objective o. Each subset updates the pheromone of the colony for which o is the *main objective*. The split procedure is described in Pseudocode 11.

```
PROCEDURE splitIntoRegion(POS)
For o in O:
        R_o = empty set
S = POS
While S is not empty:
        W = empty set
        For o in O:
                s = best solution in S w.r.t. objective o
                add s to W
                add s to R_o
```

```
remove from S all elements of W
```

Pseudocode 11. Split of the POS into region.

We introduce the following approach in order to promote the selection of updating solutions which minimise the number of residual conflicts.
In each region R_o, solutions are lexicographically sorted in worsening order[2] according to the two objectives CFL and o. The first N_{upd} elements of R_o are then selected for pheromone update. Let S_i^o be the i-th solution of R_o after sorting, and $\delta(S_i^o)$ the pheromone increment for all the edges, both in Layer 1 and Layer 2 graphs, utilised by solution S_i^o. $\delta(S_i^o)$ is computed as follows

$$\delta(S_i^o) = \frac{1}{1 + \left(CFL(S_i^o) - CFL(S_1^o)\right)} \cdot max\left(1, \frac{1}{1 + k_o\left(o(S_i^o) - o(S_1^o)\right)}\right) \qquad \text{Eq. 43}$$

k_o is equal to $+1$ in case the optimisation direction of o is minimisation, otherwise it is equal to -1. In Eq. 34 a lexicographic prioritization between the objectives CFL and o can be recognized. $\delta(S_1^o)$ is always equal to 1, while $0 < \delta(S_i^o) < 1$ for each $1 < i \leq N_{upd}$. The *max* operator in the second factor is necessary to keep the result minor or equal than 1, since the difference $\left(o(S_i^o) - o(S_1^o)\right)$ could become negative.

4 A MILP formulation for the TTP

4.1 Data model for the MILP formulation

In this section the MILP formulation used to refine solutions provided by the MOACO algorithm is introduced. We exploit the same data model and notation previously presented in Sections 2.1 and 2.2. Two main differences exist with the approach implemented for the MOACO.
On the one hand, conflict constraints are implemented as hard ones. Resulting timetables are therefore conflict-free. On the other hand, the model considers each course of the group as a single one, with its own set of variable and constraints, which are no more defined at the level of a whole train group. Courses belonging to the same group are linked by a dedicated set of constraints, called *train-linking*, which allow to properly handle periodicity tolerances.
To this purpose, we define a set of trains T, and for each train $t \in T$ let g being the train group to which t belongs, being i_t the index of t within g (having period p_g), being equal to 0 for the first train of the group. We define a journey $J_t = J_g$, and a priority factor $pr_t = pr_g$.
For each location (node) l visited along the journey J_t, we define:

- The minimum and maximum arrival times at l, $\underline{arr}_{t,l} = \underline{arr}_{g,l} + i_t \cdot p_g$ and $\overline{arr}_{t,l} = \overline{arr}_{g,l} + i_t \cdot p_g$ respectively. They are defined, only if $l \neq \underline{l}$.
- The minimum and maximum departure times from l, $\underline{dep}_{t,l} = \underline{dep}_{g,l} + i_t \cdot p_g$ and $\overline{dep}_{t,l} = \overline{dep}_{g,l} + i_t \cdot p_g$ respectively. They are defined, only if $l \neq \bar{l}$.
- The rolling stock used by the train on the edge leaving l, only if $l \neq \bar{l}$;
- The set $\mathrm{T}_{t,l} = \mathrm{T}_{g,l}$ of usable station tracks;
- The set $\Gamma_{t,l} = \Gamma_{g,l}$ of usable pass/stop modes;
- The minimum and maximum stop times $\underline{stop}_{t,l} = \underline{stop}_{g,l}$ and $\overline{stop}_{t,l} = \overline{stop}_{g,l}$.
- The minimum and maximum run times admitted for the train in the edge e leaving l (only if $l \neq \bar{l}$), $\underline{rt}_{t,e} = \underline{rt}_{g,e}$ and $\overline{rt}_{t,e} = \overline{rt}_{g,e}$;
- The linked train at a location l $lt_{t,l}$. This is the train w.r.t. which we need to impose a periodicity at l: it is the one following t in its group ($i_{lt_{t,l}} = i_t + 1$). If no linked train exists, $lt_{t,l} = t$.
- The minimum and maximum time separation w.r.t. the arrival and departure times of $lt_{t,l}$ at a location $\underline{tsa}_{t,lt_{t,l},l}$, $\overline{tsa}_{t,lt_{t,l},l}$, $\underline{tsd}_{t,lt_{t,l},l}$, $\overline{tsd}_{t,lt_{t,l},l}$. With these values we manage periodicity and periodicity tolerance as follows:
    - $\underline{tsa}_{t,lt_{t,l},l} = p_g - tol_{g,l}$;

[2] I.e. an order such that each element of the sorted set features a worse-or-equal value of the objective in point than the previous elements. If the optmization direction of this objective is minimisation, an ascending ordering is produced.

- $\overline{tsa}_{t,lt_{t,l,l}} = p_g + tol_{g,l}$;
- $\underline{tsd}_{t,lt_{t,l,l}} = p_g - tol_{g,l}$;
- $\overline{tsd}_{t,lt_{t,l,l}} = p_g + tol_{g,l}$;

From the data, we can derive a set S_e for each edge $e \in E$ including the pairs or trains which may claim e in overlapping time intervals. A pair of trains $(t, t') \in T^2$ will belong to S_e with $e = (u, v, j)$ if the index of t in T precedes the one of t', $e \in J_t \cap J_{t'}$ and

- $dir_{t,e} = dir_{t',e}$ and $\underline{dep}_{t,u} \leq \overline{dep}_{t',u}$ and $\overline{dep}_{t,u} \geq \underline{dep}_{t',u}$, or $\underline{arr}_{t,v} \leq \overline{arr}_{t',v}$ and $\overline{arr}_{t,v} \geq \underline{arr}_{t',v}$, or
- $dir_{t,e} \neq dir_{t',e}$, with t traversing u before v, and $\underline{dep}_{t,u} \leq \overline{arr}_{t',u}$ and $\overline{dep}_{t,u} \geq \underline{arr}_{t',u}$, or $\underline{arr}_{t,v} \leq \overline{dep}_{t',v}$ and $\overline{arr}_{t,v} \geq \underline{dep}_{t',v}$

Similarly, we derive set S_τ with the pairs of trains that can claim the same track τ concurrently. A pair of trains $(t, t') \in T^2$ will belong to S_τ with $\tau \in \mathrm{T}_v$ if the index of t in T precedes the one of t', $v \in J_t \cap J_{t'}$, $\tau \in \mathrm{T}_{t,l} \cap \mathrm{T}_{t',l}$ and $\underline{arr}_{t,v} \leq \overline{dep}_{t',v}$ and $\overline{dep}_{t,v} \geq \underline{arr}_{t',v}$. Finally, we define the sets $\hat{S}_e$ and $\hat{S}_\tau$ as S_e and S_τ but including the pairs of trains for which an order is implicitly defined through $\underline{arr}_{t,v}$, $\overline{arr}_{t,v}$, $\underline{dep}_{t,v}$ and $\overline{dep}_{t,v}$. If a pair of trains (t, t') is in one of these sets, it means t precedes t' there.

4.2 The MILP formulation

To model the timetabling problem, we define the following sets of non-negative continuous variables:

- Arrival time of train $t \in T$ at node $v \in J_t \backslash \underline{\mathrm{l}}_t$: $arr_{t,v} \in [\underline{arr}_{t,v}, \overline{arr}_{t,v}]$;
- Departure time of train at node $v \in J_t \backslash \overline{\mathrm{l}}_t$: $dep_{t,v} \in [\underline{dep}_{t,v}, \overline{dep}_{t,v}]$;
- Running time of train $t \in T$ on an edge $e \in J_t$: $rt_{t,e} \in [\underline{rt}_{t,e}, \overline{rt}_{t,e}]$;
- Energy consumption of train $t \in T$ on an edge $e \in J_t$: $ec_{t,e}$;

Moreover, we define the following sets of binary variables:

- For each train $t \in T$, location $v \in J_t$ and track $\tau \in \mathrm{T}_{t,l}$: $z_{t,\tau}$ equal to 1 if t uses τ, 0 otherwise;
- For each train $t \in T$ and $\varepsilon \in EEE_{t,e}$, with edge $e \in J_t$: $y_{t,\varepsilon}$ equal to 1 if t runs according to ε, 0 otherwise;
- For each edge $e \in \mathrm{E}$ and pair of trains $(t, t') \in S_e$: $p_{t,t',e}$ equal to 1 if $t \prec t'$ on e, 0 otherwise;
- For each track $\tau \in \bigcup_{v \in V} \mathrm{T}_v$ and pair of trains $(t, t') \in S_\tau$: $s_{t,t',\tau}$ equal to 1 if $t \prec t'$ on τ, 0 otherwise.

For p and s variables, the precedence symbol $\prec$ indicates two slightly different relations. When it refers to edges, in p variables, the relation concerns edge entrance times. Indeed, if two trains travel in opposite directions, then their entrance in the edge occurs at the opposite extremes. Instead, if two trains travel in the same direction, the relation considers the moment at which they leave the same node. The precedence relation on edges only imposes the impossibility of having two trains simultaneously present if they travel in the opposite directions. Differently, when the precedence concerns tracks in nodes, in s variables, the precedence relation concerns the whole duration of the presence of two trains, as a simultaneous occupation of a track is forbidden.

Finally, to measure robustness, we define two non-negative continuous variables:

- Minimum buffer time on edges mbe;
- Minimum buffer time on tracks $mb\tau$.

In addition to the input data described above, we will use m and M to indicate vary little and very large constant, respectively.

Multi-objective optimisation is handled by the MILP formulation by aggregating a set of costs into a single objective function to be minimise. With respect to the MILP formulation, we only deal with three objectives, namely TTT, EC and ST, since conflicts are treated as hard constraints and the number of scheduled trains is an input. To this purpose we define the cost of a time unit of travel time wTT and one energy consumption unit wEC. Minimum buffer times will appear in the objective function weighted through factor wST.

The objective function is the following:

$$\min \sum_{t\in T}\left(wTT\cdot\left(arr_{t,\bar{l}_t}-dep_{t,\underline{l}_t}\right)+\sum_{e\in J_t} wEC\cdot ec_{t,e}\right)-wST\cdot(mbe+mb\tau) \quad \text{Eq. 44}$$

On the one hand, the objective considers, for each train, the weighted travel time and energy consumption. On the other hand, it accounts for weighted minimum buffer time on edges and tracks. Remark that while the trin components are to be minimised, the buffer time ones are to be maximised.
This objective must be optimised while respecting several sets of constraints:
Each train must run according to exactly one Edge Extremity Event for each edge:

$$\sum_{\varepsilon\in EEE_{t,e}} y_{t,\varepsilon}=1 \ \ \forall t\in T, e\in J_t \quad \text{Eq. 45}$$

Each train must use exactly one of its available tracks in each node:

$$\sum_{\tau\in \mathrm{T}_{t,v}} z_{t,\tau}=1 \quad \forall t\in T, v\in J_t \quad \text{Eq. 46}$$

Each train cannot stay in a node less than the minimum or more than the maximum stop time:

$$dep_{t,v}-arr_{t,v}\geq \underline{stop}_{t,v} \ \ \forall t\in T, v\in J_t\backslash\{\underline{l}_t,\bar{l}_t\}, \underline{stop}_{t,v}>0 \quad \text{Eq. 47}$$

$$dep_{t,v}-arr_{t,v}\leq \overline{stop}_{t,v} \ \ \forall t\in T, v\in J_t\backslash\{\underline{l}_t,\bar{l}_t\}, \overline{stop}_{t,v}>0 \quad \text{Eq. 48}$$

Arrival and departure time of a train at a node are separated by little constant m if the train does not stop. This always holds for nodes where stops are forbidden ($\underline{stop}_{t,v}=\overline{stop}_{t,v}=0$). If the stop is optional ($\underline{stop}_{t,v}=0, \overline{stop}_{t,v}>0$) this must hold if the chosen EEE imposes a stop: the will link the EEE with the first node of the edge:

$$arr_{t,v}=dep_{t,v}-m \ \ \forall t\in T, v\in J_t\backslash\{\underline{l}_t,\bar{l}_t\}, \underline{stop}_{t,v}=\overline{stop}_{t,v}=0 \quad \text{Eq. 49}$$

$$arr_{t,v}\leq dep_{t,v}-m \ \ \forall t\in T, v\in J_t\backslash\{\underline{l}_t,\bar{l}_t\}, \underline{stop}_{t,v}=0, \overline{stop}_{t,v}>0 \quad \text{Eq. 50}$$

$$dep_{t,v}-arr_{t,v}-m\leq\left(\overline{stop}_{t,v}-m\right)\sum_{\varepsilon\in EEE_{t,e}\cap\{(s,s),(s,p)\}} y_{t,\varepsilon} \ \ \forall t\in T, v\in J_t\backslash\{\underline{l}_t,\bar{l}_t\}, e\in J_t, \underline{e}=v, \underline{stop}_{t,v}=0, \overline{stop}_{t,v}>0 \quad \text{Eq. 51}$$

By using constant m, we impose that the presence of the train is recorded for a strictly positive time in all situations. This has no actual impact for practical purposes (we can set m to half a second for example) but allows ensuring the disjunction of track utilisation by different trains.
If an optional stop can be performed at a node, then the EEE chosen for reaching and leaving the node must be coherent:

$$\sum_{e\in J_t:\bar{e}=v}\sum_{\varepsilon\in EEE_{t,e}\cap\{(s,s),(s,p)\}} y_{t,\varepsilon}=\sum_{e'\in J_t:\underline{e'}=v}\sum_{\varepsilon\in EEE_{t,e'}\cap\{(s,s),(s,p)\}} y_{t,\varepsilon} \ \ \forall t\in T, v\in J_t\backslash\{\underline{l}_t,\bar{l}_t\}, \quad \underline{stop}_{t,v}=0, \overline{stop}_{t,v}>0 \quad \text{Eq. 52}$$

At each node, the separation between the arrival and the departure times of each train t and its linked one $lt_{t,v}$ at a node are in the acceptable intervals:

$$arr_{lt_{t,v},v}-arr_{t,v}\geq \underline{tsa}_{t,lt_{t,v}} \ t\in T, v\in J_t\backslash\{\underline{l}_t\}: \ lt_{t,v}\neq t \quad \text{Eq. 53}$$

$$arr_{lt_{t,v},v}-arr_{t,v}\leq \overline{tsa}_{t,lt_{t,v}} \ t\in T, v\in J_t\backslash\{\underline{l}_t\}: \ lt_{t,v}\neq t \quad \text{Eq. 54}$$

$$dep_{lt_{t,v},v}-dep_{t,v}\geq \underline{tsd}_{t,lt_{t,v}} \ t\in T, v\in J_t\backslash\{\bar{l}_t\}: \ lt_{t,v}\neq t \quad \text{Eq. 55}$$

$$dep_{lt_{t,v},v}-dep_{t,v}\leq \overline{tsd}_{t,lt_{t,v}} \ t\in T, v\in J_t\backslash\{\bar{l}_t\}: \ lt_{t,v}\neq t \quad \text{Eq. 56}$$

Remark that we only impose this constraint for a train t if $lt_{t,v}\neq t$, i.e. if there is a train following t in the group.
For each train, the running time for an edge must be comply with the minimum technical one, also considering the track used at each extreme node:

$$rt_{t,e} \geq \sum_{\varepsilon \in EEE_{t,e}} \left(mrt^{rs_t}_{e,dir_{t,e},\varepsilon} \cdot y_{t,\varepsilon} + \sum_{\tau \in \mathrm{T}_{t,\underline{e}}} \Delta rt^{\tau}_{rs_t} \left(z_{t,\tau} + y_{t,\varepsilon} - 1 \right) + \sum_{\tau \in \mathrm{T}_{t,\overline{e}}} \Delta rt^{\tau}_{rs_t} \left(z_{t,\tau} + y_{t,\varepsilon} - 1 \right) \right) \forall t \in T, e \in J_t \quad \text{Eq. 57}$$

Here, we do the sum over all EEEs that can be used on the edge, knowing than at most one of them will be used. If an EEE ε is not used, variable $y_{t,\varepsilon}$ is set to 0 and all the elements of the sum disappear. In particular, additional run time is considered only if both the EEE and the track are used: in this case $z_{t,\tau} + y_{t,\varepsilon} - 1 = 1$, otherwise it equals 0.

For each train, the energy consumption on an edge must be coherent with the EEE is runs according to, plus the additional consumption due to the difference between the chosen and the minimum running time:

$$ec_{t,e} \geq mec^{rs_t}_{e,dir_{t,e},\varepsilon} - kec^{rs_t}_{e,dir_{t,e}} \cdot \left(rt_{t,e} - mrt^{rs_t}_{e,dir_{t,e},\varepsilon} \right) - M\left(1 - y_{t,\varepsilon}\right) \forall t \in T, e \in J_t, \varepsilon \in EEE_{t,e} \quad \text{Eq. 58}$$

If two trains t, t' may claim the same edge e concurrently traveling in the same direction, their departure times from the first node of the edge must be separated at least of the minimum headway time increased of a factor proportional to the increase of running time of the first train passing with respect to the minimum:

$$h^{t,t\prime} = mh^{rs_t,rs_{t\prime}}_{e,dir_{t,e},\varepsilon_t,\varepsilon_{t\prime}} + kh^{rs_t}_{e,dir_{t,e}} \cdot \left(rt_{t,e} - mrt^{rs_t}_{e,dir_{t,e},\varepsilon} \right) \quad \text{Eq. 59}$$

$$dep_{t',u} - dep_{t,u} \geq h^{t,t'} - M\left(3 - p_{t,t',e} - y_{t,\varepsilon} - y_{t',\varepsilon'}\right) \forall e \in E, (t,t') \in S_e, u \in J_t, \underline{e} = u \; for \; dir_{t,e}, \varepsilon \in EEE_{t,e}, \varepsilon' \in EEE_{t',e}, dir_{t,e} = dir_{t\prime,e} \quad \text{Eq. 60}$$

$$h^{t\prime,t} = mh^{rs_{t\prime},rs_t}_{e,dir_{t\prime,e},\varepsilon_{t\prime},\varepsilon_t} + kh^{rs_{t\prime}}_{e,dir_{t\prime,e}} \cdot \left(rt_{t\prime,e} - mrt^{rs_{t\prime}}_{e,dir_{t\prime,e},\varepsilon} \right) \quad \text{Eq. 61}$$

$$dep_{t,u} - dep_{t\prime,u} \geq h^{t\prime,t} - M\left(2 + p_{t,t',e} - y_{t,\varepsilon} - y_{t',\varepsilon'}\right) \forall e \in E, (t,t') \in S_e, u \in J_t, \underline{e} = u \; for \; dir_{t,e}, \varepsilon \in EEE_{t,e}, \varepsilon' \in EEE_{t',e}, dir_{t,e} = dir_{t\prime,e} \quad \text{Eq. 62}$$

In the first equation, if $t \prec t'$ ($p_{t,t',e} = 1$) and both trains use the edge them the constraint imposes the respect of the suitable minimum headway time, otherwise it is trivially satisfied. In parallel, the second constraint imposes the headway if $t' \prec t$ ($p_{t,t',e} = 0$).

If two trains t, t' may claim the same edge e concurrently traveling in opposite directions, the departure time of the second train from the first node of e it crosses must be greater than the arrival time of the first train at this same node:

$$dep_{t',v} \geq arr_{t,v} + mh_{e,dir_{t,e},v} - M\left(1 - p_{t,t',e}\right) \forall e \in E, (t,t') \in S_e, v \in J_t, \overline{e} = v \; for \; dir_{t,e}, \quad dir_{t,e} \neq dir_{t\prime,e} \quad \text{Eq. 63}$$

$$dep_{t,u} \geq arr_{t\prime,u} + mh_{e,dir_{t\prime,e},u} - M \cdot p_{t,t',e} \; \forall e \in E, (t,t') \in S_e, u \in J_t, \underline{e} = u \; for \; dir_{t,e}, \quad dir_{t,e} \neq dir_{t\prime,e} \quad \text{Eq. 64}$$

As for the previous couple of constraints, the first one is relevant if $p_{t,t',e} = 1$ and the second one if $p_{t,t',e} = 0$.

For each track τ, for each pair of train t, t' that may claim it concurrently, we impose that the track occupation is not overlapping, also considering the minimum separation time:

$$arr_{t',v} \geq dep_{t,v} + tg_v - M\left(3 - s_{t,t',\tau} - z_{t,\tau} - z_{t',\tau}\right) \forall v \in V, \tau \in \mathrm{T}_{t,v} \cap \mathrm{T}_{t',v}, (t,t') \in S_\tau dep_{t',v} \geq arr_{t,v} + mh_{e,dir_{t,e},v} - M\left(1 - p_{t,t',e}\right) \forall e \in E, (t,t') \in S_e, v \in J_t, \overline{e} = v \; for \; dir_{t,e}, \quad dir_{t,e} \neq dir_{t\prime,e} \quad \text{Eq. 65}$$

$$arr_{t,v} \geq dep_{t\prime,v} + tg_v - M\left(2 + s_{t,t',\tau} - z_{t,\tau} - z_{t',\tau}\right) \forall v \in V, \tau \in \mathrm{T}_{t,v} \cap \mathrm{T}_{t',v}, (t,t') \in S_\tau \quad \text{Eq. 66}$$

The minimum buffer time on edges *mbe* is the shortest time separating the entrance of two trains on the same edge if they travel in the same direction, minus the minimum headway time.

$$mbe \leq dep_{t',u} - dep_{t,u} - h^{t,t'} + M\left(3 - p_{t,t',e} - y_{t,\varepsilon} - y_{t',\varepsilon'}\right) \; \forall e \in E, (t,t') \in S_e, u \in J_t, \underline{e} = u \; for \; dir_{t,e}, \varepsilon \in EEE_{t,e}, \varepsilon' \in EEE_{t',e}, dir_{t,e} = dir_{t\prime,e} \quad \text{Eq. 67}$$

$$mbe \leq dep_{t',u} - dep_{t,u} - h^{t,t'} + M\left(2 - y_{t,\varepsilon} - y_{t',\varepsilon'}\right) \ \forall e \in E, (t,t') \in \hat{S}_e, u \in J_t, \underline{e} = u \ for \ dir_{t,e}, \varepsilon \in EEE_{t,e}, \varepsilon' \in EEE_{t',e}, dir_{t,e} = dir_{t\prime,e} \quad \text{Eq. 68}$$

$$mbe \leq dep_{t,u} - dep_{t\prime,u} - h^{t\prime,t} - M\left(2 + p_{t,t',e} - y_{t,\varepsilon} - y_{t',\varepsilon'}\right) \forall e \in E, (t,t') \in S_e, u \in J_t, \underline{e} = u \ for \ dir_{t,e}, \varepsilon \in EEE_{t,e}, \varepsilon' \in EEE_{t',e}, dir_{t,e} = dir_{t\prime,e} \quad \text{Eq. 69}$$

If the trains travel in opposite direction, the relevant times are their entrance and exit at the same location:

$$mbe \leq dep_{t',v} - arr_{t,v} - mh_{e,dir_{t,e},v} + M\left(1 - p_{t,t',e}\right) \forall e \in E, (t,t') \in S_e, v \in J_t, \overline{e} = v \ for \ dir_{t,e}, dir_{t,e} \neq dir_{t\prime,e} \quad \text{Eq. 70}$$

$$mbe \leq dep_{t',v} - arr_{t,v} - mh_{e,dir_{t,e},v} \forall e \in E, (t,t') \in \hat{S}_e, v \in J_t, \overline{e} = v \ for \ dir_{t,e}, dir_{t,e} \neq dir_{t\prime,e} \quad \text{Eq. 71}$$

$$mbe \leq dep_{t,u} - arr_{t',u} - mh_{e,dir_{t',e},u} + M \cdot p_{t,t',e} \ \forall e \in E, (t,t') \in S_e, u \in J_t, \underline{e} = u \ for \ dir_{t,e}, \quad dir_{t,e} \neq dir_{t\prime,e} \quad \text{Eq. 72}$$

Remark that Eq. 68 and Eq. 71 concern the pairs of trains that use the same edge but for which the utilisation order is defined in the input data, due to the minimum and maximum arrival and departure times at the extreme nodes.

Through p and s variables, this formulation uses disjunctive constraints to guarantee the respect of capacity constraints on tracks, following the modelling principles of RECIFE-MILP (Pellegrini et al., 2015). However, the two models consider different representations of the infrastructure, which here is macroscopic while it is microscopic in RECIFE-MILP. Here, we associate arrival and departure times to nodes rather than specific tracks. This explains higher complexity of the disjunctive constraints presented in equations Eq. 59 to Eq. 72, in which we need to ensure no relation is imposed between times if trains use alternative tracks.

A set of valid inequalities, also used in by Mannino et al. (2015) for railway scheduling, may be implemented for strengthening the model:

$$p_{t,t',e} \leq p_{t,t',e'} \ \forall e, e' \in J_t, (t,t') \in S_e \cap S_{e'}, dir_{t,e} \neq dir_{t',e}, \underline{e}' \prec \underline{e} \ for \ dir_{t,e} \quad \text{Eq. 73}$$

They exploit the observation that the precedence relation between trains traveling in opposite direction propagates along their route. In particular, if two trains t and t' use a sequence of edges including e, and t precedes t' on one of e, the t is the first to pass also on all edges e' it crosses before arriving to e: $p_{t,t',e} = 1 \Longrightarrow p_{t,t',e'} = 1$. From the opposite perspective, if t' passes first in e', then it is also first in e, as for it e precedes e': $p_{t,t',e\prime} = 0 \Longrightarrow p_{t,t',e} = 0$.

4.3 Integration into the ATMO framework

Solutions S generated by MOACO are then refined through the solution of a MILP formulation. It takes as input the train groups to be scheduled, and passing and stopping times at all locations visited in their journey. By exploiting the periodicity tolerance of trains belonging to the same group and dropping temporal discretization, the formulation slightly modifies these times and seeks for a conflict-free solution which optimises the weighted sum of the normalised values in $[\tau_{min}, \tau_{MAX}]$ of three objectives: TTT, EC, ST. Indeed, the number of trains is constant, as all train groups scheduled in S are to be scheduled here. Moreover, conflicts are modelled by hard constraints, making their minimisation meaningless. The weights considered in the formulation objective function are those used by the ant that built solution S.

The MILP formulation is allowed to search for improvements just in a neighbourhood of the MOACO solutions. This means that bounds are set to the formulation's variables, and these bounds are, in principle, stricter than those considered by the MOACO algorithm. We call them "neighbourhood bounds". In the formulation we have two types of variables:

- Binary variables control the use of specific tracks at stations and of specific EEEs, as well as the precedence between pairs of trains using the same resource.
- Continuous variables control arrival and departure times, as well as running times and energy consumption.

In the implementation so far developed, we set "neighbourhood bounds" to arrival and departure time variables only. They are bounded to be chosen within thin intervals around the ones fixed in S. The maximum allowed time modification is a parameter that we call MILP degree of freedom (DOF_{MILP}). By varying the value of continuous variables, the formulation can:

- Shift a whole periodic train group of a maximum value DOF_{MILP};
- Modify the schedule of individual trains provided that its group's periodic pattern is relaxed within the tolerance permitted in the service concept.

At now, we do not set any "neighbourhood bound" to binary variables. As it will be recalled in Section **Errore. L'origine riferimento non è stata trovata.** ("Implementation Plan"), future research should focus also in setting proper bounds to these variables.
The restricted variable domains, allow (in principle) for a fast solution. This process returns either the optimal solution or an infeasibility due to the modelling of conflicts as hard constraints.

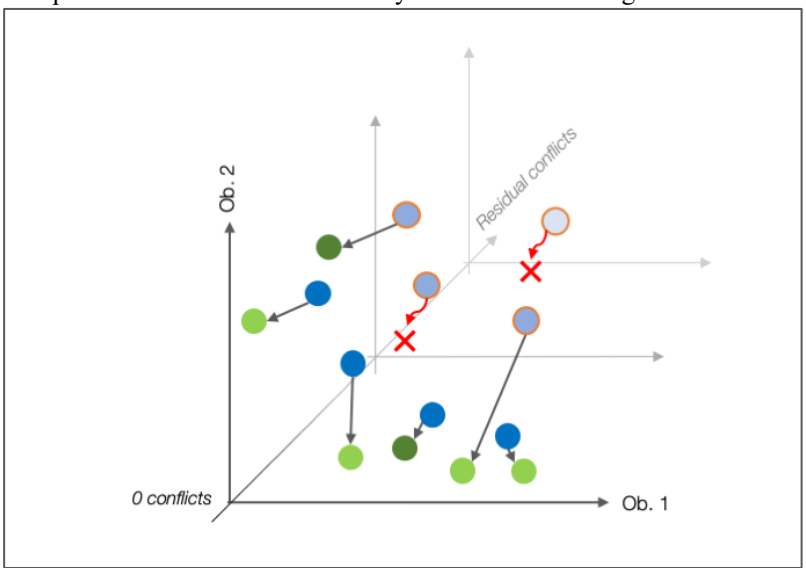

Figure 4. Graphical representation of the MILP formulation operation in the objectives space.

Figure 4 provides a graphical representation of the operation of the MILP formulation stage in the objective function space. Two objectives are considered here. MOACO solutions are blue points. Some are conflict-free, other not (the red-circled ones). Dark blue points represent the POS provided by MOACO. Green points are MILP solutions obtained starting from MOACO ones. Not all MOACO solutions with residual conflict can be converted into a conflict free timetable (red crosses). A new POS is originated by the MILP solutions. As the MILP stage only refines the MOACO solutions, without adding new ones, this final POS may at most contain the same number of solutions as the final MOACO POS, but likely less, as some refined solutions may dominate others. Light green points represent MILP solutions which belong to this new POS, while dark green points do not.
As anticipated in Section 2.3, in this research we investigate the application of the MILP formulation only for refining the solutions of the POS produced by the MOACO algorithm. The study of how to effectively use the MILP formulation as a Local Search to be run within the MOACO's iterations is left to future research activities.
POSs produced by MOACO could likely feature a significant cardinality. Furthermore, many of these timetables would likely present a high number of residual conflicts, so that they could return unfeasibility when trying to refining them through the MILP formulation. Applying the MILP formulation to all the MOACO POS timetables would therefore result in a significant waste of time, since for real-size instances also the construction of the MILP model and demonstration of unfeasibility take significant time, of the order of some tens of seconds, 1 minute maximum. With MOACO POS with cardinality of several tens, time waste becomes significant. In order to contain it, we implement the following procedure:

1. Solutions of the MOACO POS are sorted in ascending order w.r.t. the number of residual conflicts;
2. A counter i_{UNF} is initialised to 0;
3. Sorted solutions are iteratively refined with the MILP formulation: if this operation returns an unfeasibility, i_{UNF} is incremented by 1;
4. If i_{UNF} is greater than a given threshold n_{UNF}, the iterative refinement is stopped.

In the presented application, we empirically set n_{UNF} to 5. Future research should focus on a more deepened definition of this threshold, as well as on possible different procedures to manage the application of the MILP formulation.

5 Applications and results

5.1 Test instances

Tuning and tests are performed on 54 instances drawn from real timetabling practice in Norway. Instances are built on two parts of the Norwegian national network, namely the Bergen line (Bergensbanen, BB) and the region around Trondheim (Trønderbanen, TRB). These areas form a good test-bench for our framework under working conditions. Both are realistic instances of regional strategic timetable projects, as well as realistic medium-sized assignments to individual timetable planners when partitioning the national network as part of a network-wide project. While this partitioning helps overcome the complexity of the TTP, it easily leads to a sub-optimisation of the network-wide timetable, for human planners and ATMO alike.

Table 2. Main characteristics of the test infrastructure

Infrastructure	BB	TRB
Length	370	200
Signalling system	Conventional Norwegian signalling, based on axel-counters	
Number of timing locations	53	52
Of which stations	32	30
Average n° of available sidings in stations	2.3 (excluding the 8-tracks Bergen station)	2.4 (excluding the 13-tracks Trondheim station)

Tuning and tests are performed on 54 instances drawn from real timetabling practice in Norway. Instances are built on two parts of the Norwegian national network, namely the Bergen line (Bergensbanen, BB) and the region around Trondheim (Trønderbanen, TRB). These areas form a good test-bench for our framework under working conditions. Both are realistic instances of regional strategic timetable projects, as well as realistic medium-sized assignments to individual timetable planners when partitioning the national network as part of a network-wide project. While this partitioning helps overcome the complexity of the TTP, it easily leads to a sub-optimisation of the network-wide timetable, for human planners and ATMO alike.
Table 2 provides a concise overview of the main infrastructure characteristics of these lines, while Figure 5 frames them within the whole Norwegian railway network.

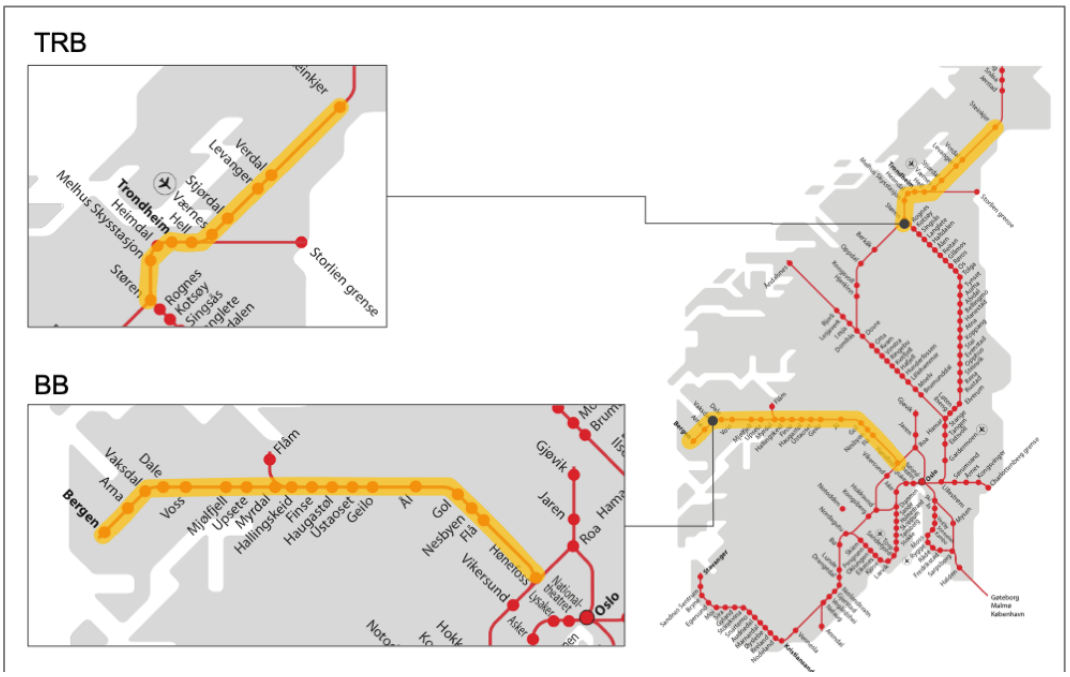

Figure 5. Geographical framing of the considered lines.

On each infrastructure, different instances are built using a reference daily traffic pattern, whose main characteristics are described in Figure 5 and Table 3. For each considered train category (FR: freight; LH: long-haul passenger; R: regional) are reported: the number of trains in the two directions; the period, just in case relevant trains have to be scheduled with a periodic pattern; the number of stations in which trains can stop and use a passing loop; the number of such stations in which a stop is mandatory, the remaining ones being optional stops, where a passing is normally preferred.

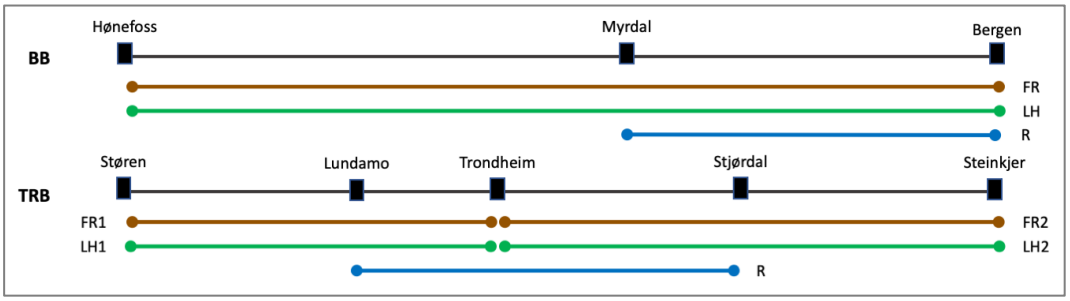

Figure 6. Base traffic on the test infrastructures.

Table 3. Main characteristics of the main base traffic patterns.

TRB					
Category	FR1	FR2	LH1	LH2	R
N° trains	5+5	5+5	8+7	6+7	15+16
Period (min)	-	-	-	60	60
N° usable crossing stations	5	20	6	21	16
Of which mandatory stops	0	0	6	16	16
Max stop time (min)	60	60	8	8	5
BB					
Category	FR		LH		R
N° trains	8+7		5+5		15+15
Period (min)	-		60		60
N° usable crossing stations	26		21		21
Of which mandatory stops	0		21		21

Max stop time (min)	60	8	5

From the base traffic patterns, we derive 24 and 30 test instances for the TRB and the BB line, respectively. These are produced by combining:

- Four different sizes of the admissible time windows of train group timings in stations (the difference between $\overline{arr}$ and $\underline{arr}$ (as well as $\overline{dep}$ and $\underline{dep}$) in each location of a train's journey): 15, 30, 60 and 90 minutes (for the TRB line only, also 120 minutes);
- Six different active train configurations, in which 0 to 5 train groups (randomly chosen in the traffic patterns) are deactivated and removed from the scheduling process.

The input datasets of the instances were prepared using the Treno suite of railway timetable planning software (see Medeossi and Nash, 2020) currently used by Jernbanedirektoratet. A prototype dedicated interface provides seamless data transfer between Treno and the ATMO framework and vice-versa.

All the 54 resulting instances are tested activating the objectives *TTT* and *ST*, in addition to the default *CFL*. As no train group is optional, the maximisation of the number of scheduled trains is not meaningful here. As for energy consumption, we excluded it because input data are not currently available in the necessary level of detail.

5.2 MOACO assessment

5.2.1 Tuning

The proposed MOACO features a remarkable number of parameters, which can hardly be tuned manually. The tuning will be performed by means of the IRACE tool (López-Ibáñez et al., 2011). It automatically selects the best configuration of continuous, discrete and cardinal parameters among those defined as an input, considering a set of reference problem instances. This is an open-access state-of-the-art tuning procedure, which can be applied both to single and multi-objective problems. It is based on advanced machine learning techniques.

The parameters setting for the tuning procedure is reported in Table 4. Discrete sets of possible parameters configurations are described as comma-separated lists, while continuous ranges are reported in square brackets. A bold font highlights the values selected by IRACE. A fixed time limit of 500 s a *tuning budget* (the maximum number of algorithm's invocation that IRACE can exploit to perform the tuning) of 5000 iterations are set. The hypervolume HV (to be maximised, Fonseca et al., 2006) is chosen as the comparison KPI. With these settings, the tuning procedure takes almost 12 days to be completed, on a Intel(R) Xeon(R) CPU E5-2637 v3 @ 3.50GHz with 16 CPUs and 125 GB ram.

Table 4. Settings and results of the IRACE tuning procedure. "int" annotation highlights intervals defined in $\mathbb{N}$, otherwise intervals are defined in $\mathbb{R}$.

$Nant$	20, **35**, 50	$n_{SMART} : n_{FAST}$	0:1, **1:9**, 3:7, 5:5, 7:3, 9:1, 1:0
α_{L1}	[0, 10] *int* **3.0**	$n_{LS} : n_{noLS}$	**0:1**, 1:0, 1:1, 1:2, 1:4
β_{L1}	[0, 10] *int* **3.0**	$pcSol_{LS}$	25, 50, 75, 100
ρ_{L1}	[0.01, 0.1] **0.092**	N_{upd}	1, 2, **3**
$\tau_{MIN,L1}$	[0.05, 1.0] **0,467**	Fixed Parameters	
$\tau_{MAX,L1}$	[6.0, 12.0] **8.0**	$maxTime$	500 s
α_{L2}	[0, 10] *int* **2.0**	$N^{weights}$	5
β_{L2}	[0, 10] *int* **8.0**	$\underline{\lambda}$	0.5
ρ_{L2}	[0.01, 0.1] **0.070**		
$\tau_{MIN,L2}$	[0.05, 1.0] **0.618**		
$\tau_{MAX,L2}$	[6.0, 12.0] **12.0**		

The tuning highlighted that the alternation between the smart and fast TEG exploration modes actually improves the algorithm's performances, with an optimal ratio of 1:9 respectively. The optimal value for N_{upd} is equal to 3. This means that at the end of each iteration, 3 different solutions contribute to update the pheromone trails of each colony. This implies that a differentiation during the exploration of the solutions' space is fostered. On the other side, it emerged that the

proposed Local Search strategy does not improve the global performances of the algorithm, since the tuning decides to not exploit it. This fact will be further investigated in the following.

5.2.2 Comparisons

In this section further comparisons between different parameters configurations are presented. We considered the following configuration:

- *BEST*: The parameters configuration returned by the tuning;
- *GRASP*: This configuration implements a Greedy Randomized Adaptive Search Procedure (GRASP) obtained from BEST by setting the α parameters to 0 (in both L1 and L2), while the other ones are left unchanged. In this way we exclude the pheromonal memory from contributing guiding the exploration during each iteration. Exploration will therefore be guided (in a randomized way) by the heuristic information only.
- *noHeurL1*: The dynamic computation of the heuristic factor in Layer 1 is a computationally onerous operation. However, the tuning highlights it produces a performances improvement. In order to further investigate these behaviours, a configuration in which the dynamic computation of the Layer 1 heuristics is turned off (setting $\beta_{L1} = 0$) is tested.
- *BEST+LS:* The tuning chooses not to exploit at all the proposed Local Search strategy. However, the 5th best configuration identified by IRACE in its internal ranking takes advantage of the Local Search with n_{LS} : n_{noLS} equal to 1:9 and $pcSol_{LS}$ equal to 25. This configuration is therefore tested.

For each configuration, all the 54 test instances are run, with a random set of seeds. A time limit of 500 s is set. We then compare pairs of configurations by applying the Wilcoxon test to the differences between the HVs returned, for the same instance, with different configurations. The Wilcoxon signed-rank test is a non-parametric statistical hypothesis test used to compare two populations using a set of matched samples. It returns whether the two samples sets belong to different populations (i.e. they are statistically different) and if the considered KPI of one set is statistically greater (or minor) than the other, with a given confidence level. This test can therefore return if one configuration will "on average" perform better than the other for all the items of a hypothetic set from which the test instances are drawn. The confidence level used in our tests is 5 %.
First of all, we compare the configurations BEST and GRASP. In this way we would asses the performances of the whole MOACO algorithm. It is expected that the MOACO algorithm performs better than the GRASP one, and the comparison is intended to measure the benefits of the extra "artificial intelligence" provided by pheromone.
Figure 7 reports the values of the HV differences ($HV_{BEST} - HV_{GRASP}$), sorted in ascending order, for the 54 instances. Positive values characterise instances for which the MOACO outperforms the GRASP. The Wilcoxon test returns that the MOACO performs statistically better than the GRASP.

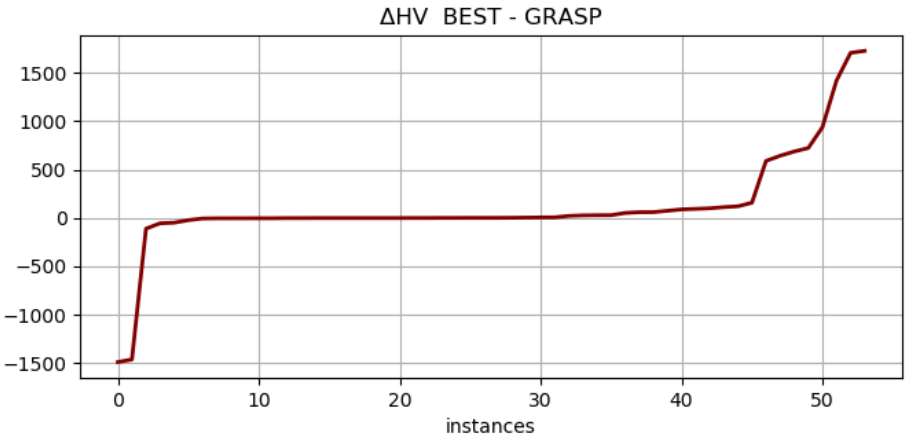

Figure 7. HV difference between the BEST and GRASP configurations for the 54 test instances.

Figure 8 reports the number of iterations performed, within the same 500 s time limit, with the considered parameters configurations. With reference to the BEST configuration, an average relative difference of +1.89 %, + 15.82 % and -48.97 % in the number of performed iterations is obtained

for configurations GRASP, noHeurL1 and BEST+LS respectively. For the BEST configuration, the number of performed iterations spans between 190 and 4530.

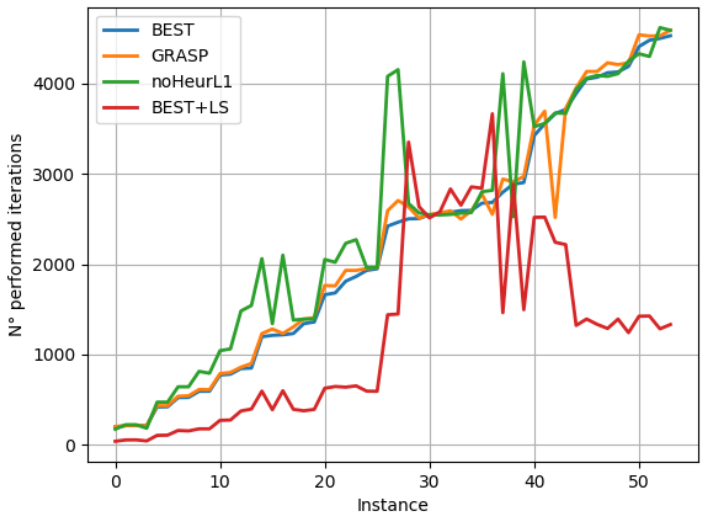

Figure 8. Number of performed iterations within the 500 s time limit.

Figure 9 reports the HV differences between configurations noHeurL1, BEST+LS and BEST. The Wilcoxon test applied to these distributions revealed that both noHeurL1 and BEST+LS perform worse than BEST, thus confirming the tuning outputs. We hypothesize that the significant overhead computation time required by the Local Search prevents the algorithm to perform a sufficient number of iterations necessary to exploit at best the pheromonal memory.

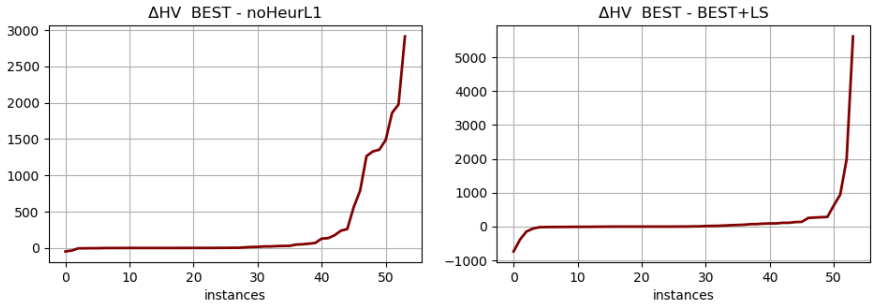

Figure 9. HV difference between the BEST and noHeurL1 and BEST+LS configurations for the 54 test instances.

5.2.3 MILP refinement

In this section, we report the results obtained when refining with the MILP formulation the timetables of the 54 POSs obtained with the *BEST* configuration. To this purpose, DOF_{MILP} is set to 20 minutes and the periodicity tolerance of all groups in all relevant locations is set to 10 minutes. The MOACO is run until a 500 s time limit is met, and then the MILP formulation is solved considering each timetable of the produced POS as input. The MILP formulation is solved by the GUROBI 9.1 commercial solver. To comparison purposes, we run two sets of experiments, refining the same set of timetables produced by the MOACO using two termination criteria for the MILP formulation, namely TC1 and TC2:

- TC1: the solver stops when either a 2% MIP gap or a 600 s time limit are met;
- TC2: the solver stops when either a 2% MIP gap or a 900 s time limit are met.

In this way we aim at assessing the influence of the MILP computation time to the quality and quantity of produced solutions.

Table 5. Overview of main results from the application of the MILP formulation to the 54 test instances with TC1.

#	min CFL	max CFL	ns ACO	ns MILP	t MILP (min)	#	min CFL	max CFL	ns ACO	ns MILP	t MILP (min)
1	0	0	3	3	0,8	*28*	0	23	5	5	10
2	0	9	7	1	0,9	*29*	0	179	13	12	10
3	0	84	9	9	1,1	*30*	0	11	6	1	10
4	0	42	9	9	1,2	*31*	0	47	9	9	10
5	0	39	5	2	1,2	*32*	0	136	14	14	10
6	90	130	5	5	1,2	*33*	0	179	15	15	10
7	0	15	5	2	1,4	*34*	0	18	6	1	10
8	66	130	8	8	1,4	*35*	0	126	11	8	10
9	0	24	9	9	1,7	*36*	0	34	8	8	10
10	11	24	7	7	2,3	*37*	2	205	23	14	10
11	3	114	26	15	2,4	*38*	0	40	11	11	10
12	32	137	13	13	2,4	*39*	70	407	19	6	10
13	0	95	19	15	2,7	*40*	33	116	13	4	10
14	2	172	20	13	3,6	*41*	121	404	23	5	10
15	8	104	19	5	3,9	*42*	102	455	28	0	N/F
16	6	213	23	12	4,3	*43*	1	9	6	0	N/F
17	10	11	2	2	7,2	*44*	3	9	6	0	N/F
18	0	23	6	6	8,9	*45*	54	106	11	0	N/F
19	0	0	5	5	9,0	*46*	155	330	24	0	N/F
20	0	0	2	2	9,5	*47*	35	84	11	0	N/F
21	0	0	3	2	10	*48*	61	479	23	0	N/F
22	0	0	4	4	10	*49*	150	463	20	0	N/F
23	0	0	4	1	10	*50*	1	8	7	0	?
24	0	42	6	6	10	*51*	92	508	24	0	?
25	36	159	11	9	10	*52*	35	103	21	0	?
26	4	106	23	10	10	*53*	1	10	9	0	?
27	1	153	19	10	10	*54*	242	406	18	0	?

Table 5 reports the results obtained for the 54 instances, with termination criterion TC1. For each instance, columns *min CFL* and *max CFL* report the minimum and maximum numbers of residual conflicts in the timetables produced by the MOACO. Columns *ns MOACO* and *ns MILP* report the cardinality of the POS before and after the MILP refinement, respectively. The difference between these two values indicates the number of timetables for which a conflict-free solution cannot be found by solving the MILP formulation. Column *t MILP* indicates the average computation time in minutes per timetable required to solve the MILP formulation, calculated only for those timetables for which a feasible solution is found.

With TC1, for 41 of the 54 instances, the MILP formulation computes at least one conflict-free timetable. For 17 of these 41 instances, the 2% MIP gap condition is met for all the MILP-refined timetables, while for the remaining 24 the MILP solver find a feasible solution but stops after the 600 s without reaching the 2 % MIP gap. The MILP formulation does not provide any feasible timetable for 13 instances: for 8 of them, unfeasibility is attested (marked as "N/F" in column *t MILP*), meaning that no feasible timetable exists in the neighbourhood of the solutions produced by MOACO. For the remaining 5, the MILP solver is not capable to verify whether the model is feasible or not within the 600 s time limit (marked as "?" In column *t MILP*).

Not all the timetables produced by the MILP refinement belong to a not-dominated POS: on average, 27 % of refined timetables are dominated and, as such, discarded.

Table 6. Overview of main results from the application of the MILP formulation to the 54 test instances with TC2.

#	min CFL	max CFL	ns ACO	ns MILP	t MILP (min)	#	min CFL	max CFL	ns ACO	ns MILP	t MILP (min)
1	0	0	3	3	0,8	*28*	0	23	5	5	14,3
2	0	9	7	1	0,9	*29*	0	179	13	12	14,5
3	0	84	9	9	1,1	*30*	0	11	6	3	14,8
4	0	42	9	9	1,2	*31*	0	47	9	9	15
5	0	39	5	2	1,2	*32*	0	136	14	14	15
6	90	130	5	5	1,2	*33*	0	179	15	15	15
7	0	15	5	2	1,4	*34*	0	18	6	3	14,7
8	66	130	8	8	1,4	*35*	0	126	11	8	14,8
9	0	24	9	9	1,7	*36*	0	34	8	8	15
10	11	24	7	7	2,3	*37*	2	205	23	14	15
11	3	114	26	15	2,4	*38*	0	40	11	11	15
12	32	137	13	13	2,4	*39*	70	407	19	6	15
13	0	95	19	15	2,7	*40*	33	116	13	7	15
14	2	172	20	13	3,6	*41*	121	404	23	11	15
15	8	104	19	5	3,9	*42*	102	455	28	0	N/F
16	6	213	23	12	4,3	*43*	1	9	6	0	N/F
17	10	11	2	2	7,2	*44*	3	9	6	0	N/F
18	0	23	6	6	10,2	*45*	54	106	11	0	N/F

19	0	0	5	5	10,5	*46*	155	330	24	0	N/F
20	0	0	2	2	13,4	*47*	35	84	11	0	N/F
21	0	0	3	2	13,2	*48*	61	479	23	0	N/F
22	0	0	4	4	13,6	*49*	150	463	20	0	N/F
23	0	0	4	2	14,2	*50*	1	8	7	2	15
24	0	42	6	6	14	*51*	92	508	24	0	?
25	36	159	11	10	14,1	*52*	35	103	21	0	?
26	4	106	23	12	14,8	*53*	1	10	9	2	15
27	1	153	19	12	14,8	*54*	242	406	18	0	?

Table 6 reports the same set of indicators, computed with termination criterion TC2. The instances ordering in Table 6 is the same as in Table 5, so that, for example, instance #3 is the same in the two sets of experiments. The table highlights how, in general, by allowing more 5 minutes more, the solver is capable to return a higher number of conflict-free timetables. Blue lines in Table 6 highlight those instances for which the number of conflict-free timetables obtained with TC2 is major than that obtained with TC1. Furthermore, for two of the instances for which with TC1 no feasible timetables could be found but at the same time model unfeasibility could not be assessed, with TC2 two feasible timetables can be computed.

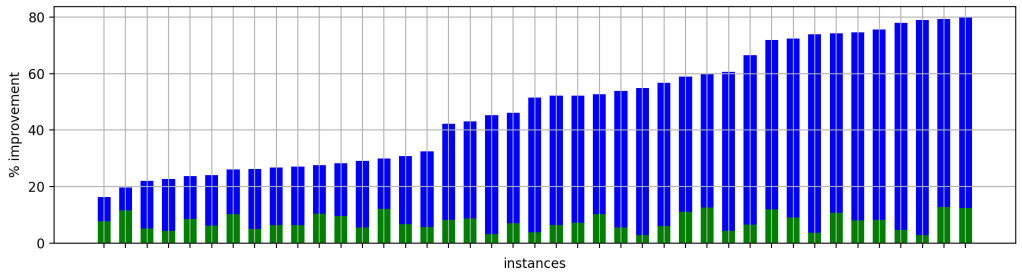

Figure 10. Improvement in the values of objectives TTT (blue bars) and ST (green bars) produced by the MILP refinement with TC1.

Figure 10 describes the average percentage improvement in the objective functions' values produced by the MILP refinement, for those instances for which at least one feasible timetable is found with TC1. Blue bars refer to improvement of the TTT objective, while green bars to the ST one. The figure highlights how dramatic improvements are produced by the MILP formulation, thanks to the exploitation of periodicity tolerances. Indeed, it is well-known that the enforcement of rigid periodic patterns is a severe capacity constraint mainly for operations on single track (Emery, 2010).
When comparing percentual improvements obtained with TC1 and TC2, it can be pointed out that only minor improvements (on average on all instance, of the order of 3.5 %) in the value of both the objective functions can be achieved by lengthening the MILP solver's computation times. It can be inferred that, at least in the analysed instances, the main and significative effect of computation times prolongation is the greater number of feasible timetables that can be produced.

5.3 Application case study 1: Assessment of infrastructure variants on the Bergen Railway

In this section the ATMO is applied to an infrastructure planning case study exercise on the BB infrastructure. The case study considers the 370 km single-track railway line between Bergen and Hønefoss. The existing infrastructure is designated as scenario zero (S0) and two infrastructure improvement scenarios are defined each of which added 50 km of double track along the line. In scenario 1 (S1) three double track sections are placed equally spaced along the line. In scenario 2 (S2) a single long double track stretch is placed approximately in the middle of the line.
The case study objective is to develop conflict-free timetables that maximises the number of freight trains that could be scheduled in a 24-hours day (starting at midnight) and minimises additional total travel time required due to crossings, given a specified passenger train timetable. Therefore, two algorithm objectives are activated, namely minimisation of additional travel time and maximisation of number of scheduled optional trains. This illustrates how the tool's multi-objective approach enables planners to easily study the relationships between multiple objectives (in this case two objectives).
Three traffic patterns are defined in the case study:

- FR: line capacity should be saturated with up to 37 freight trains.
- LH+FR: 10 quasi-periodic long haul passenger trains (5 in each direction) must be scheduled; the remaining capacity should be saturated with up to 37 freight trains.
- R+LH+FR: In addition to the long-haul passenger trains, 30 periodic regional trains (15 in each direction) must be scheduled following a clock-face headway pattern between Bergen and Myrdal in the time window 7:00 AM to 10:00 PM, the remaining capacity should be saturated with up to 37 freight trains. This traffic pattern features 77 trains and 16547 train·km.

In addition to the base traffic patterns, the case study also tests the influence of periodicity tolerances in the periodic pattern of passenger trains. To this purpose the following two sets of traffic patterns are constructed to be overlaid on the base one:

- "X" pattern where strict periodicity is required (no periodicity tolerance).
- "T" pattern where a tolerance of ±15-minutes for long-haul trains and ±5-minutes tolerance for regional trains is allowed at all stations.

The infrastructure alternatives and service concepts were are for the ATMO tool using the Treno suite of railway timetable planning software (see Medeossi and Nash, 2020) currently used by Jernbanedirektoratet. A dedicated interface provides seamless data transfer between Treno and the ATMO tool and vice-versa. The infrastructure scenarios as well as the base service concepts are illustrated schematically in Figure 11.

Freight trains are set as optional, this means the algorithm can choose whether to schedule them or not. This enables the ATMO to search for the best trade-off between number of scheduled trains and additional travel time.

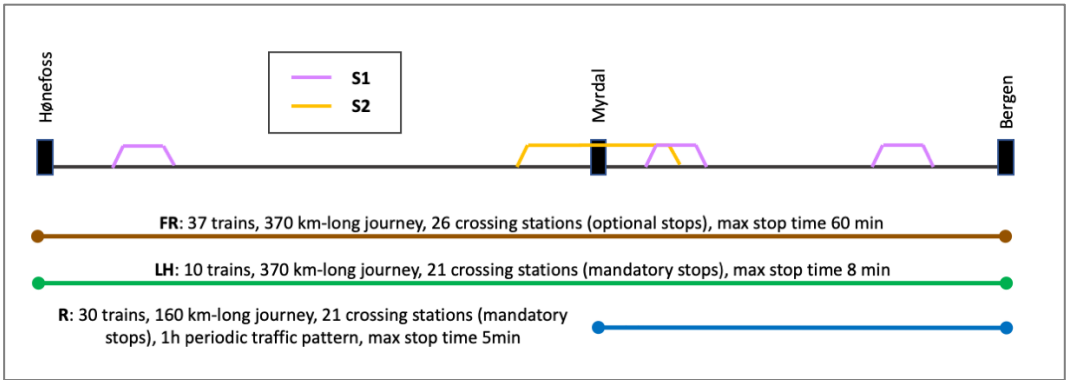

Figure 11. Case study infrastructure and service concepts.

The combination of three infrastructure alternatives and six service concepts leads to 18 scenarios. The ATMO tool is applied to all 18 scenarios. The following termination criteria are set to the algorithmic framework. The MOACO algorithm is stopped after a 30-minute time limit. Then the MILP formulation is applied to all the timetables in the provisional POS. The MILP formulation is solved using the GUROBI commercial solver, which stops when a 2% MIP gap (an indicator which measures how far the current solution is from its estimated optimum) or a 5-minute time limit is met. In the 85 % of the experiments for which the MILP formulation returns a feasible solution, the MIP gap termination condition is met. Differently from the test described in Section 5.2, experiments are carried out on a MacBook Pro with 2.6 GHz Intel Core i7 6 core processor and 16 GB RAM. The choice of this machine intends to provide a snapshot of the algorithm's performances closer to real practice.

Figure 12 presents the results as approximations of the Pareto fronts relevant to the X scenarios (no periodicity tolerance allowed). The points composing the fronts (lines) represent actual conflict-free timetables produced by the algorithm. Each point on the line quantifies the key performance indicators (KPIs) selected for this case study, namely the number of scheduled freight trains and the total additional travel time due to crossings. Each point on a line is Pareto optimal, in other words it is the maximum possible number of freight trains for the given additional travel time.

The lines display the best trade-off between the objectives (KPIs) for each infrastructure / service concept scenario. This means that timetables falling under and to the right of the Pareto front lines can be designed, but do not use capacity optimally (more freight trains could be operated with the same total delay). Similarly, timetables falling in the sector over and to the left of the Pareto front lines cannot be designed without causing traffic conflicts. In particular, the figure shows that an

upper bound exists for the number of freight trains which it is possible to schedule in each scenario, no matter how much additional travel time is scheduled.

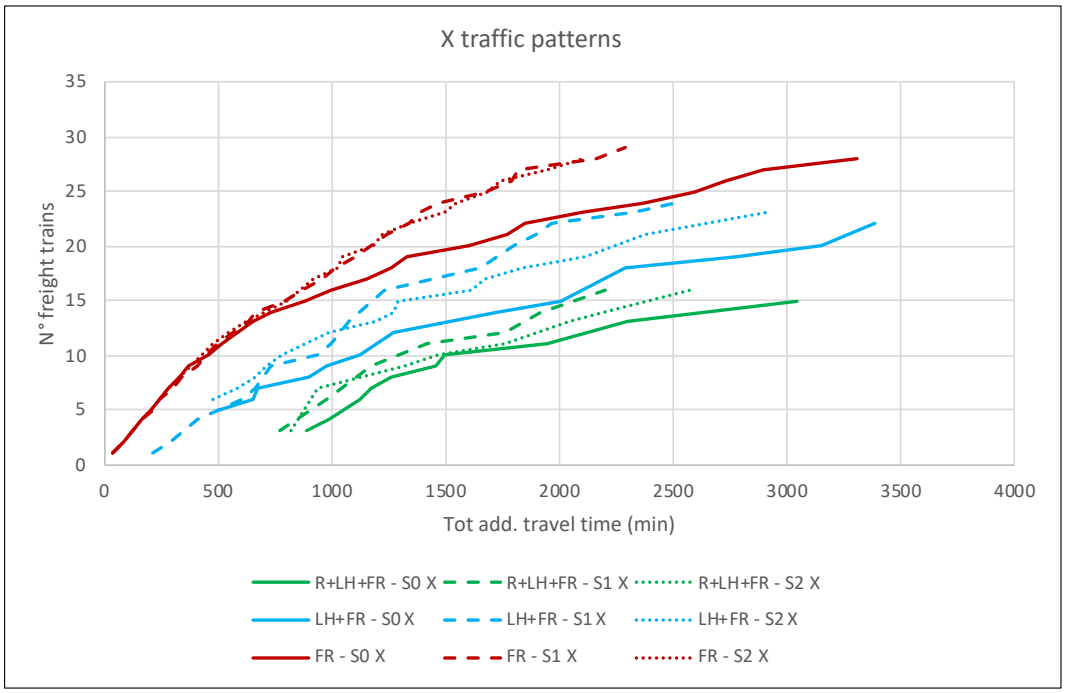

Figure 12. Pareto Fronts Representing Capacity Utilisation Trade-off for Analysed Scenarios. Colors refer to the three service concepts considered in the case study. The solid lines represent the infrastructure baseline S0, the dashed lines represent alternative S1, and the dotted lines alternative S2.

In terms of strategic planning the results presented in Figure 12 can be used to make the following conclusions regarding the case study:

- The difference between infrastructure scenarios S1 and S2 only arises when passenger trains are scheduled. In this case, S1 performs better than S2. When only freight trains are considered, there is no difference between S1 and S2.
- The presence of passenger trains strongly affects the maximum number of freight trains that can be scheduled. In LH+FR scenarios, the maximum number of schedulable freight trains is 18.8 % lower than in the FR scenarios (based on 3 FR scenario average). In R+LH+FR scenarios, this percentage increases to 44.7 %. Or, looked at another way, total additional travel time increases significantly when passenger trains are added to timetable with a specified number of freight trains.
- Adding double track increases the number of freight trains that can be operated under all service scenarios for a given additional travel time. This is due to reducing the number of stops for crossings, which can now be performed in double track stretches. In particular, considering the upper-right extremities of the Pareto fronts, in scenarios S1 an average travel time reduction of 36.3 % can be achieved compared to scenarios S0 (average value over the three service concepts). For scenarios S2, the average reduction is equal to 26.8 %.

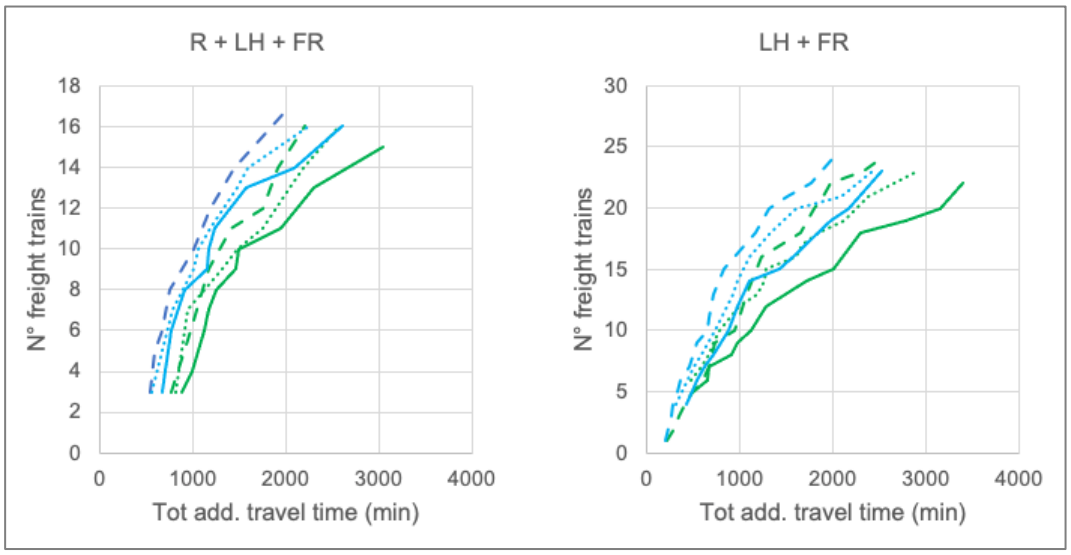

Figure 13. Influence of Periodicity Tolerances. Green lines refer to the X traffic pattern (strict periodicity) and blue to the T traffic patterns (with periodicity tolerances). Solid lines refer to the S0 baseline, the dashed lines represent S1, and the dotted lines represent the S2 infrastructure alternatives.

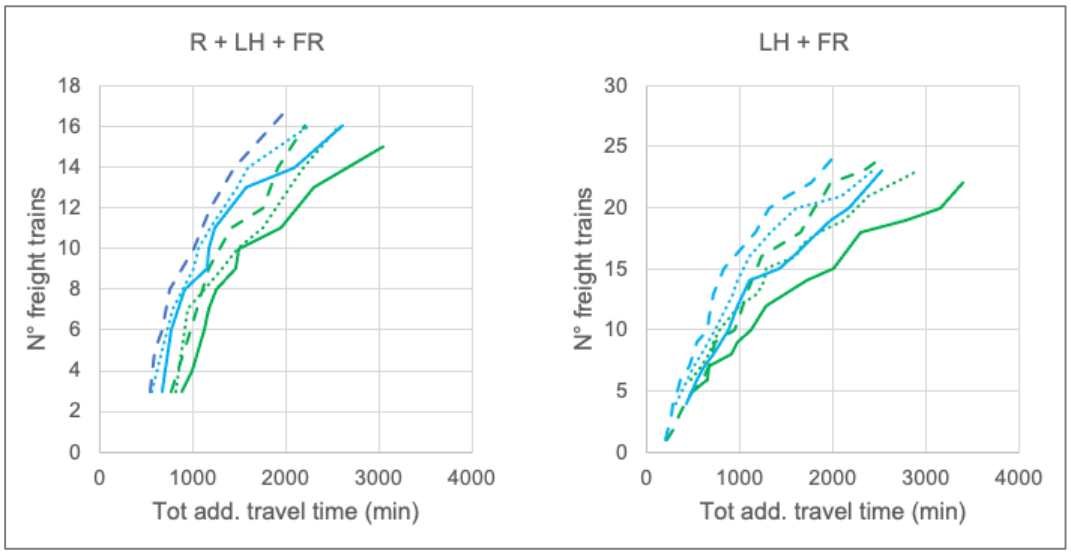

Figure 13 illustrates the influence of periodicity tolerances in the passenger train traffic patterns ("T" scenarios). As shown, allowing more flexibility over a rigid periodic pattern is an effective measure for reducing the time losses due to crossings. The general findings for the infrastructure alternatives are the same both in "X" and "T" scenarios.

This simple case study highlights the benefits of the new ATMO tool. Using a multi-objective optimisation and Pareto fronts, it provides planners with much more information than a classic timetabling approach which produces just one timetable at time.

Furthermore, the tool also can produce these timetables efficiently. Table 7 illustrates the computation times required to produce the relevant Pareto front as well the number of timetables returned as an output for each scenario analysed in the case study. Columns *# TT tot* report the total number of generated timetables, while columns *# TT POS* how many of them belong to the actual POS for each scenario. The figure shows how the use of a multi-objective metaheuristic combined with a MILP-based refinement can be used to produce a significant number of timetables in a relatively small computation time. In total, the case study required approximately 13.6 hours of computation time (it can be run overnight) and produced 822 timetables.

Table 7. Number of calculated timetables and computation times for each scenario.

scenario	S0			S1			S2		
	# TT tot	# TT POS	c.t. (min)	# TT tot	# TT POS	c.t. (min)	# TT tot	# TT POS	c.t. (min)
FR	37	28	32,2	76	29	56	70	28	35,7
R+LH+FR X	53	10	107,2	34	8	68,5	37	7	74,9
LH+FR X	41	13	39,1	126	18	52,7	55	15	38
R+LH+FR T	30	10	62,3	59	8	66,1	38	8	67,3
LH+FR T	27	13	37,9	68	18	35,2	71	14	44,3

In summary, the case study showed how the ATMO tool can be used to improve the strategic planning process by efficiently creating many feasible timetables for comparing infrastructure – service concept scenarios. As illustrated in the case study, the ATMO tool results, although based on aggregated KPIs, are effective as a first step during strategic capacity analysis of timetable planning. They provide planners with a quick overview of the main trends for the KPIs
The results also help planners identify problem areas to be studied in detail. More specifically, each point on a POS line represents an actual timetable that can be further analysed using microscopic timetable-planning software. This detailed study would use the standard methods, i.e., an iterative loop of manual adjustments and traffic simulations, to design-test-analyse the specific issues of interest.

5.4 Application case study 2: Passenger traffic scheduling on the Western Oslo node

The purpose of this application case study is to assess the computational limitation of the ATMO tool. To this purpose, we apply it to a computationally "hard" TTP instance, that is the scheduling of a half-day of passenger railway traffic on the Western Oslo Node. The choice of scheduling a half-day only derives from previous experiments, which pointed out that the entire day is totally too onerous from a computational perspective to be tackled by the current version of the tool.

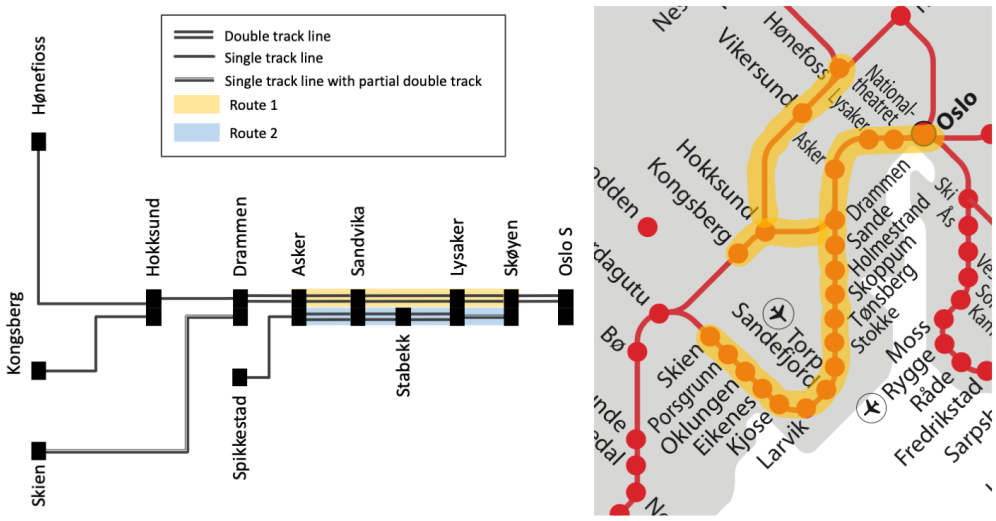

Figure 14

The considered infrastructure is schematically represented in Figure 14. It is composed by two sets of lines:

- The two west-bound double-track lines exiting the Oslo Sud station, which run in an approximately parallel way and merge in Asker station. The southern line features several intermediate halts, while the northern one is designed to provide a more direct route to Asker. We therefore define two routes for trains running between Oslo Sud and Asker: Route 1 on the northern line, Route 2 on the southern one.
- The single-track lines (some of them featuring partial double track stretches) Asker-Spikkestad, Drammen-Skien, Drammen-Hønefoss and Hokksund-Kongsberg.

The main features of the service concept to be scheduled on this infrastructure are reported in Table 8. For train group, identified by its two termina stations, the number of courses of the two directions as well as the used route between Oslo S and Asker are reported. As anticipated, we consider only courses departing from the first station of their journey in the time slot 00:00 – 12:00. All these train groups are modelled as movable ones, with fixed stopping pattern (no optional stops allowed). Maximum stop times and periodicity tolerances are defined separately for two sets of stations:

- Stations belonging to the two routes (between Asker, comprised, and Oslo S): maximum stop time equal to 2 minutes, periodicity tolerance equal to 2 minutes;
- All the other stations: maximum stop time equal to 6 minutes, periodicity tolerance equal to 10 minutes.

Table 8. Main characteristics of the service concept.

Train group	N° courses	Route
Oslo S-Skien	6 / 11	1
Oslo S-Asker	12 / 12	2
Oslo S-Asker	6 / 0	1
Oslo S-Stabekk	31 / 30	2
Oslo S-Skøyen	6 / 0	0
Oslo S-Kongsberg	8 / 11	1
Oslo S-Hønefoss	2 / 0	1
Oslo S-Drammen	41 / 48	1
Oslo S-Spikkestad	12 / 12	2

This service concept features 248 courses and 20775 train·km, and it is deemed to be a sufficiently "hard" TTP instance to be solved by the ATMO tool. With this input dataset, a set of experiments have been run, aiming at understanding where the limitations of the current version of tool lay, from both a practical and a theoretical point of view.
To this purpose, we define 6 test configurations by setting different values of the time limit to the MOACO algorithm and to the Gurobi solver tackling the MILP formulation. Table 9 describes these configurations. The 2% MIP gap termination criterion is used for all test configuration too.

Table 9. Test configurations.

# Test	Time limit MOACO (s)	Time limit MILP solver (s)
1	1800	1800
2	1800	3600
3	3600	1800
4	3600	3600
5	7200	1800
6	7200	3600

The target of the experiments is to analyse how the quantity and quality of the solutions is affected by the limits on the computation time. For each test configuration, Table 10 reports the number of iterations performed by MOACO (*n Iters ACO*), the size of the POS provided by MOACO (*ns ACO*) as well as by the MILP refinement (*ns MILP*). Furthermore, column *min CFL ACO* indicates the number of residual conflicts present in the MOACO POS timetable with the best value of this objective, while column *best TTT ACO* displays the value of the TTT objective for this timetable. Finally, column *best TTT MILP* reports the value of the TTT objective of the best timetable (according to TTT) present in the POS returned by the MILP refinement. The last column indicates the total computation time required to run each configuration test.

Table 10. Results obtained for the test configurations.

# Test	n Iters ACO	ns ACO	ns MILP	min CFL ACO	best TTT ACO	best TTT MILP	el.t. (h)
1	436	6	0	286	214593	-	3,4
2	889	7	0	298	215145	-	4,5
3	874	11	0	256	213565	-	6,5
4	902	9	5	263	208165	14906	8,3
5	1673	19	4	240	239725	13805	8,5
6	1725	15	10	239	196405	13174	9,1

Results highlight how the Western Oslo Node actually represents a hard TTP instance to be tackled with the ATMO tool. In facts, comparing the current ratios of computation time and number of provided with those of the Bergen Railway application case study (Section 5.3), it can be noticed that they are now dramatically significantly higher.
Furthermore, with configurations 1,2,3 the MILP formulation is not capable to refine any of the timetables produced by the MOACO. In these cases, the time limit set for the MILP solver is reached before any feasible timetable is calculated, without, on the other side, demonstrating unfeasibility. It can be inferred that a 1800 s time limit is not sufficient for the MOACO algorithm to provide timetables sufficiently optimised to be easily refined by the MILP formulation. With a 3600 s MOACO time limit, feasible solutions can be found by the MILP solver if a 3600 s time limit is set to the latter.
Finally, when letting the MOACO algorithm run for 7200 s, in 1800 s the MILP solver is capable to compute feasible timetables. If a time limit of 3600 s is allowed for the MILP solver, the quantity and the quality (in terms of the TTT value of the best POS solution, column *best TTT MILP*) of the provided solutions are significantly improved.
These tests confirmed that the current version of the ATMO tool must be further improved in terms of performances to efficiently tackle large size instance as the Western Oslo Node. Furthemore, it has been pointed out how the blending of the MOACO algorithm and the MILP formulation (+ solver) has to be properly tuned to achieve sufficient quality and quantity of solutions while containing the computation times. An outlook of possible ways to achieve these improvements are presented in the conclusions.

6 Conclusions and future research potential

This document describes a novel algorithmic framework for automatic railway timetable generation, integrated into the ATMO tool prototype. Thanks to its multi-objective approach, it is suited to perform strategic timetabling and timetable-based capacity analysis, since it provides the user with sets of Pareto-optimal timetables according to 4 different KPIs, thus matching the requirements set by the Norwegian Railway Directorate for the present project.
It has been described how the ATMO algorithmic framework is composed by the integration of an original MOACO algorithm and a MILP formulation, focusing on the features of the MOACO algorithm which constitutes, to the best of our knowledge, an original contribution to the field of solutions methods for the TTP.
A series of application cases are discussed, highlighting on the one hand the current ATMO performances are adequate to effectively solve TTP instances with up to 70 daily trains to be scheduled on (mainly) single track lines. On the other hand, the Western Oslo Node instance, with more than 2 hundred trains to be scheduled in a 12-hours' time window proved to be too demanding for the current version of the tool, putting in evidence the need for further improvements.
In facts, the ATMO tool produced as an output of the present project and its algorithmic framework should be regarded as still at an early-stage development phase. In facts, it is suitable to be improved in different ways, which are presented in the following.

Local Search

Numerical tests highlighted that the proposed Local Search strategy designed to improve the solutions produced by ants actually fails to do that, possibly because of its significant computational overhead. This calls for the development of alternative heuristics to be used as Local Search. Several techniques have been assessed in ACO-related literature, but none of them deals explicitly with the solution of the TTP. This was the main reasons behind the choice of developing the dedicated Local Search procedure described in this deliverable. Efforts must be performed in adapting effective Local Search procedures from literature.
The usage of the MILP formulation as Local Search throughout the MOACO could be an effective measure, and must be properly studied. Since the computation time to solve the MILP formulation would be higher than that required by a simpler heuristic, a proper application criterion must be carefully designed. On the other side, the MILP model is actually different from that used by the MOACO, because of the hard conflict constraints and of the periodicity tolerances. This implies that the variables' spaces of MOACO and MILP are different. For this reason, once a MOACO solution is refined by the MILP formulation, it can be compared with MOACO ones in the objectives' space only, but no more in the variables' space. This fact must be carefully addressed while implementing the usage of MILP as a Local Search.

Progressive Refinement

The MOACO algorithm could be improved by introducing a progressive refinement of its model, following the approaches proposed by.
Walshaw (2004) and Blum et al. (2008) proposed the so-called *progressive refinement* techniques for improving the performances of meta-heuristic optimisation algorithms. A progressive refinement is performed when a problem can be represented by a set of models (in case of ACO they are the construction graphs) of increasing granularity or resolution. In principle, given a constant size of the search space, a lower resolution implies a lower number of variables. On the one hand, a "coarse" model can in principle be explored faster than a high-resolution one. On the other hand, an optimal solution on a coarse model will be sub-optimal in a high-resolution one, but an optimal solution on a high-resolution level is likely to lay in a neighbourhood of an optimal solution of a low-resolution one.
With this approach, the algorithm (whatever it is, being genetic, tabu-search, ACO, etc.) starts searching an optimal solution on the coarser level, then it swaps to a more refined one but restricting the search in a neighbourhood of the already found optimal solution. The procedure continues until the most refined level is reached. Notice that in the proposed ATMO framework this approach is already implicitly implemented, since the MILP formulation model is basically a refined version of the MOACO ones (time as a continuous variable, periodicity tolerances). Within the very MOACO, this approach can be explicitly implemented in two ways:

1. by progressively reducing the groups' time discretization;
2. by allowing some trains to be virtually extracted from their strictly periodic group, explicitly modelling the periodicity tolerances used by the MILP formulation in the MOACO algorithm too.

Dedicated research should define and test proper application criteria for these two techniques, considering the active objectives. For instance, refinement could be applied to the most conflictive trains (groups) in order to help the MOACO to produce solutions with a lower number of residual conflicts.

Conflicts and stability KPIs

In this study the KPI for the objective CFL (Minimisation of conflicts) is the number of conflicts themselves. A possible alternative could be to compute the total size of the overlap between the concurrent utilisations slots on the same infrastructure resource which result into a conflict. In this way, two conflicts, one with a "large" overlap, the other with a "little" overlap, will count in different ways in the objectives' computation. With the current version, their contributions are absolutely the same. It can be hypothesised that the utilisation overlap KPI would likely help to MOACO to produce solutions for which the MILP formulation can find a feasible timetable in an easier way.
In this study stability is modelled by means of the *a-priori* ((according to Goverde and Hansen, 2013)) KPI "minimum buffer time in the timetable", to be maximised. To this purpose, many other KPIs could be certainly designed, provided that they can be integrated into the sequential solution construction implied by the MOACO. Further research should focus on a comparison between possible different stability KPIs. Microsimulation of the generated timetables can be used to get *a-posteriori* stability indicators.
It is worthwhile to point out that the overlap KPI can be extended to act as an a-priori stability KPI, since the overlap is basically a "negative buffer time" (and can be computed like a buffer time). This observation paves the way for future possible utilisation of just one single objective for CFL and ST.

Neighbourhood for MILP refinement

In this study, the neighbourhood of the MOACO solutions to be explored by the MILP formulation is defined for time variables only. Other variables (station tracks, EEEs) are let free to vary regardless of the choices performed in the MOACO solution. Future research should focus on an effective definition of a neighbourhood for these variables too, in order to reduce MILP computation times, possibly without major quality detriment of solution quality.

Bulk timetables analysis

Application experiments highlighted how the ATMO returns the user with a potentially high number of timetable variants. This number would further dramatically increase if the user is interested not only on the Pareto Optimal Set of solutions, but in all (or at least a part) the feasible solutions produced during the algorithm's iterations. Such a number of timetables can be exploited to perform

analysis based on timetable patterns recognition which could provide more insight on the ways capacity can be used. Methods to massively analyse whole sets of timetables should therefore be developed, opening a novel (to the best of our knowledge) direction in the field of Railway Operation Research.

References

1. Stützle T., Hoos H., 1997. Improvements on the Ant System: Introducing the MAX-MIN Ant System. In G. Smith, N. Steele and R. Albrecht (eds.), Procs. of Artificial Neural Nets and Genetic Algorithms, pp.245–249.
2. Hansen, I.A., 2000. Station capacity and stability of train operations. Computers in Railways VII. 7. 809-816.
3. Stützle T., Hoos H., 2000. MAX − MIN Ant System. In Journal of Future Generation Computer Systems, vol. 10, pp.889–914.
4. Delorme X., Rodriguez J., Gandibleux X., 2001. Heuristics for railway infrastructure saturation. Electronic Notes in Theoretical Computer Science 50(1), 39-53.
5. Walshaw C., 2004. Multilevel refinement for combinatorial optimisation problems. Annals of Operation Research volume 131, 325-372.
6. UIC, 2004. Code 406: Capacity, Editions Techniques Ferroviaires, Paris.
7. Fonseca C. M., Paquete L., Lopez-Ibanez M., 2006. An Improved Dimension-Sweep Algorithm for the Hypervolume Indicator. 2006 IEEE International Conference on Evolutionary Computation, 1157-1163.
8. Solnon C., Bridge D., 2006. An Ant Colony Optimisation Meta-Heuristic for Subset Selection Problems. In System Engineering using Particle Swarm Optimisation, Nova Science Publisher, 7-29.
9. Abril M., Barber F., Ingolotti L.P., Salido M.A., Tormos M.P., Lova A., 2008. An assessment of railway capacity. Transportation Research Part E 44(5), 774-806.
10. Blum C., Yábar Vallès M., Blesa M.J., 2008. An ant colony optimisation algorithm for DNA sequencing by hybridization, Computers & Operations Research, 35(11), Issue 11, 3620-3635.
11. Kontaxi, E. & Ricci, S., 2009. Techniques and methodologies for carrying capacity evaluation: comparative analysis and integration perspectives. Ingegneria Ferroviaria 64(12), 1051-1080.
12. Emery D., 2010. “Increasing the capacity of a single-track line”, proceedings of the 10th Swiss transport research conference 2010.
13. Lindner T., 2011. Applicability of the analytical UIC Code 406 compression method for evaluating line and station capacity. Journal of Rail Transport Planning & Management (2011), 49-57.
14. López-Ibáñez M., Dubois-Lacoste J., Stützle T., Birattari M., 2011. The irace package, Iterated Race for Automatic Algorithm Configuration. Technical Report TR/IRIDIA/2011-004, IRIDIA, Université libre de Bruxelles, Belgium.
15. López-Ibáñez M., Stützle T., 2012. The automatic design of multiobjective ant colony optimisation algorithms. IEEE transactions on evolutionary computation, 16(10).
16. Nothegger C., Mayer A., Chwatal A., Raidl G. R., 2012. Solving the post enrolment course timetabling problem by ant colony optimisation. Annals of Operations Research volume 194, 325–339.
17. Goverde R. M. P. and Hansen I. A., 2013. Performance indicators for railway timetables. 2013 IEEE International Conference on Intelligent Rail Transportation Proceedings, Beijing, 2013, pp. 301-306, doi: 10.1109/ICIRT.2013.6696312.
18. KFH Group, 2013. Transit capacity and quality of service manual. Washington.
19. Landex A., Jensen L.W., 2013.Measures for track complexity and robustness of operation at stations. Journal of Rail Transport Planning & Management, 3(1-2), 22-35.
20. UIC, 2013. Code 406 R: Capacity, Editions Techniques Ferroviaires, Paris.
21. Coviello N., 2014. Modelling periodic operations on single track lines: Timetable design and stability evaluation, Research in Transportation Economics, special issue on Rail Operations, Management and Economics, Volume 54, December 2015, 2-14.

22. De Fabris S., Longo G., Medeossi G., Pesenti R., 2014. Automatic generation of timetables based on a mesoscopic infrastructure model. Journal of Rail Transport Planning & Management, 4(2014), 2-13.
23. Hansen I.A., Pachl J., 2014. Railway timetabling & operations. Hamburg: Eurailpress.
24. Yao B., Hu P., Zhang M., Tian X., 2014. Improved Ant Colony Optimisation for Seafood Product Delivery Routing Problem. PROMET - Traffic&Transportation. 26.
25. Pellegrini P., Marlière G., Pesenti R., Rodriguez J., 2015. RECIFE-MILP: An effective MILP-based heuristic for the real-time railway traffic management problem. IEEE Transactions on Intelligent Transportation Systems 16(5):1-11.
26. Mannino C., Lamorgese L., Piacentini M., 2015. Optimal train dispatching by benders'-like reformulation. Transportation Science, 50(3): 763-1138.
27. Pouryousef H., Lautala P., 2015. Hybrid simulation approach for improving railway capacity and train schedules. Journal of Rail Transport Planning & Management. 5(4).
28. Pouryousef H., Lautala P., White T., 2015. Railroad capacity tools and methodologies in the US and Europe. Journal of Modern Transportation 23(1), 30-42.
29. Cacchiani, V., Furini, F. & Kidd, M., 2016. Approaches to a real-world train timetabling problem in a railway node. Omega 58, 97-110.
30. Samà M., Pellegrini P., D'Ariano A., Rodriguez J., 2016. Ant colony optimisation for the real-time train routing selection problem. Transportation Research Part B, 85(3), 89-108.
31. Goverde R.M.P., Bešinović N., Binder A., Cacchiani V., Quaglietta E., Roberti R., Toth P., 2016. A three-level framework for performance-based railway timetabling. Transportation Research Part C: Emerging Technologies, Volume 67, 62-83.
32. Caimi G., Kroon L., Liebchen C., 2017. Models for railway timetable optimisation: Applicability and applications in practice. Journal of Rail Transport Planning & Management, 6(4), 285-312.
33. Coviello N., Pellegrini P., Sobieraj S., Rodriguez J., 2017. Stability of saturated timetables: the influence of buffer times. In 7th International Conference on Railway Operations Modelling and Analysis - RailLille2017, Lille, France, 486–504.
34. Khoshniyat F., Törnquist Krasemann J., 2017. On-Demand Timetabling in Dense Railway Networks: Methods and Challenges. 7th International Conference on Railway Operations Modelling and Analysis - Rail Lille.
35. Jensen L.W., Landex A., Nielsen O.A., Kroon L.G., Schmidt M., 2017. Strategic assessment of capacity consumption in railway networks: Framework and model. Transportation Research Part C: Emerging Technologies, 74(2), 126-149.
36. Lamorgese L., Mannino C., Natvig E., 2017. An exact micro–macro approach to cyclic and non-cyclic train timetabling, Omega, Volume 72, 59-70.
37. Pellegrini P., Marlière G., Rodriguez J., 2017. RECIFE-SAT: A MILP-based algorithm for the railway saturation problem. Journal of Rail Transport Planning & Management 7(1-2), 19-32.
38. Schupbach, K., Caimi, G., & Jordi, J. (2017). Towards Automated Capacity Planning in Railways.
39. Bešinović N., Goverde R.M.P., 2018. Capacity Assessment in Railway Networks. In Borndörfer, Klug, Lamorgese, Mannino, Reuther, Schlechte (Eds.), Handbook of Optimisation in the Railway Industry. International Series in Operations Research & Management Science, 268, Springer, Cham (2018).
40. Broman E., Eliasson J., Aronsson M., 2019. A Mixed Method for Railway Capacity Allocation. In RailNorrköping 2019. 8th International Conference on Railway Operations Modelling and Analysis (ICROMA), Norrköping, Sweden, June 17th–20th, 2019 (No. 069, pp. 482-490). Linköping University Electronic Press, 232-245.
41. Jordi J., Toletti A., Caimi G., Schüpbach, 2019. Applied Timetabling for Railways: Experiences with Several Solution Approaches. In RailNorrköping 2019. 8th International Conference on Railway Operations Modelling and Analysis (ICROMA), Norrköping, Sweden, June 17th–20th, 2019 (No. 069, pp. 462-470). Linköping University Electronic Press.
42. Liebchen C., Schülldorf H., 2019. A collection of aspects why optimisation projects for railway companies could risk not to succeed–A multi-perspective approach. Journal of Rail Transport Planning & Management 11.
43. Peterson A., Polishchuk V., Schmidt C., 2019. Applying Geometric Thick Paths to Compute the Maximum Number of Additional Train Paths in a Railway Timetable. In RailNorrköping 2019. 8th International Conference on Railway Operations Modelling and

Analysis (ICROMA), Norrköping, Sweden, June 17th–20th, 2019 (No. 069, pp. 984-997). Linköping University Electronic Press.
44. Weik N., Hemminki E., Nießen N., 2019. The effective residual capacity in railway networks with predefined train services. Operations Research Proceedings 2019, 752-731.
45. Medeossi G., Nash A., 2020. “Reducing Delays on High-Density Railway Lines: London–Shenfield Case Study”, Transportation Research Record: Journal of the Transportation Research Board, 2674 (7), pp. 193-205.
46. Weymann, F.; Nießen, N.: Optimisation processes to assist with fine compilation of timetables. In: ETR Internation Edition 1 (2015) 1, S. 24 – 27
47. Meurer, D.; van Hövell, M.; Büker, Th.: "Kapazitätsermittlung für Teilnetze unter Variation von Routing und Infrastrukturszenarien". In: ETR 4(21), pp. 24-29., https://www.via-con.de/wp-content/uploads/024_029_Meurer_Hoevell_Bueker_FA.pdf