Periodic Railway Timetabling with Event Flexibility*

Gabrio Caimi, Martin Fuchsberger, Marco Laumanns, and Kaspar Schüpbach

Institute for Operations Research, ETH Zurich, 8092 Zürich, Switzerland
{caimig,fumartin,laumanns}@ifor.math.ethz.ch, kaspasch@student.ethz.ch

Abstract. This paper addresses the problem of generating conflict-free periodic train timetables for large railway networks. We follow a two level approach, where a simplified track topology is used to obtain a macro-level schedule, and the detailed topology is considered locally on the micro level. To increase the solution space in the interface of the two levels, we propose an extension of the well-known Periodic Event Scheduling Problem (PESP) such that it allows to generate flexible time slots for the departure and arrival times instead of exact times. This Flexible Periodic Event Scheduling Problem (FPESP) formulation considerably increases the chance to obtain feasible solutions (exact train routings) subsequently on the micro level, in particular for stations with dense peak traffic. Total trip time and the time slot sizes are used as multiple objectives and weighted and/or constrained to allocate the flexibility where it is most useful. Tests on a medium size instance of the Swiss Federal Railways 2007 service intention demonstrate the advantage of the FPESP model, while it only moderately increases its solution time in most cases.

1 Introduction

Railway traffic in Europe has increased considerably for both passenger and freight transportation, and this trend is expected to continue. As construction of new tracks is very expensive and hardly possible in many city centers, it is crucial to utilize the existing infrastructure as good as possible to meet the customer demand for an enlarged offer. With increasing density of the timetable, however, scheduling trains becomes more and more difficult not only with respect to safety restrictions, but also for mitigating propagation of delays. The prospect of automatic generation of conflict-free timetables in reasonable time is therefore considered very promising by railway companies in the production as well as in the planning phase, here in order to evaluate several alternative timetables.

* We thank the Swiss Federal Railways for funding and providing data and in particular Dr. Felix Laube, Samuel Roos, Oskar Stalder, and Dr. Raimond Wüst for insightful discussions. Furthermore, we are grateful to Dr. Fabian Chudak and Dr. Leon Peeters for fruitful discussions we had at the very beginning of this work. Finally, we thank the referees for their constructive comments that helped improving the quality of this paper.

ATMOS 2007 (p.124-141)
7th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems
http://drops.dagstuhl.de/opus/volltexte/2007/1173

The Swiss Federal Railways Infrastructure Division (SBB-I), for instance, major operator of the railway infrastructure in Switzerland, is currently investing efforts into the development of efficient methods for generating and operating railway schedules [9, 16, 26].

Our research focuses on the construction of periodic timetables for a given *train service intention*, which describes the train services that passenger and freight companies would like to offer. This train service intention consists of train lines with frequencies and specifies customer-relevant information such as stop stations, interconnection possibilities, and train type. The goal is to create detailed train schedules, which specify an exact itinerary through the railway topology with passing times for each train. This way the provided timetable is guaranteed to be conflict-free, i.e., assuming no delays, all trains can run exactly as planned without creating safety conflicts. This feature is in contrast to today's timetables, which are typically not planned to be conflict-free and rather rely on on-line resolution of resource conflicts as they occur in real time.

As it appears intractable to consider the detailed topology all at once, we propose a two-level approach for generating conflict-free train schedules [2]. In the macroscopic (or macro) level, given a train service intention for the whole railway network, we abstract from the detailed track topology for creating a draft timetable. In the microscopic (or micro) level, starting with the draft timetable from the macro level, we construct detailed train schedules by considering locally precise topologies, the corresponding safety system as well as accurate train dynamics. For micro scheduling, several models and algorithms are available for solving large problems with many trains and routing possibilities [27, 5, 6, 1].

This paper focuses on the periodic timetabling on the macroscopic level. This can be modeled as a Periodic Event Scheduling Problem (PESP, see [11]) whose output (departure and arrival times) serves as the input for the micro level to check feasibility by finding a feasible routing. Our goal is to increase the chance for finding a feasible routing on the microscopic level. We reach this goal by generalizing the PESP model to search for arrival and departure time intervals in lieu of exact event times, which are quite restrictive for the micro level and often lead to infeasibility. This additional flexibility for those events leads to the extended model developed in this paper, the *Flexible Periodic Event Scheduling Problem* (FPESP).

Other methods for generating non-periodic train schedules consider a simplified topology for a line [3] or a larger network, applying a heuristic that sequentially fixes the train sequence [4] or use a multicommodity flow approach [23]. However, the importance of the periodicity for timetables in Switzerland as well as results in the Netherlands [24] and in Germany [10] suggest that the PESP is a powerful model for coping with macroscopic train timetabling.

This paper is organized as follows: In Section 2 we discuss the PESP and give a literature review on the relevant work on this model. Section 3 contains the main contribution of the paper, the introduction of flexibility for the events in the PESP model. Section 4 presents computational results on a test case in central Switzerland, and in Section 5 we give an outlook for future research.

2 The classical PESP model and literature review

This section introduces the Periodic Event Scheduling Problem, a powerful model for periodic schedules introduced by Serafini and Ukovich [25] which was first applied to train scheduling by Schrijver and Steenbeck [24].

2.1 Classical PESP model

A periodic railway schedule on the macro level consists of a list of departure and arrival times at the nodes (stations) in the aggregated network for all trains running within an hour. Each departure or arrival of a train at a node is called an *event* i which takes place at a certain time π_i. As the schedule is periodic with a time period T (often $T = 60$ min), the event i also takes place at times $\{\ldots, \pi_i - T, \pi_i, \pi_i + T, \pi_i + 2T, \ldots\}$. Therefore, π_i can be restricted to $0 \leq \pi_i < T$.

The choices of the event times π_i depend on each other. For instance, two trains running on the same track cannot have the same departure times. These dependencies are modeled as constraints in the PESP. The constraints always concern two events i and j and define the minimum and maximum periodic time difference l_{ij} and u_{ij} between the two. The constraint bounds l_{ij} and u_{ij} are given as data of the model, and scheduling is then about finding event times π_i for each event i that fulfill all constraints of the form

$$l_{ij} \leq \pi_j - \pi_i + Tp_{ij} \leq u_{ij}. \tag{1}$$

The integer variables p_{ij} allow the constraints to be fulfilled in the periodic sense. As an example, Eq. (1) with $l_{ij} = 10$, $u_{ij} = 15$, and $T = 60$ can be fulfilled by $\pi_i = 46$, $\pi_j = 58$, and $p_{ij} = 0$ but also by $\pi_j = 1$ where $p_{ij} = 1$ enables the jump to the next time period.

The events and constraints constitute the elements of the *Periodic Event Scheduling Problem* (PESP). This problem can be solved by the corresponding integer linear program (ILP) formulation [11, 22, 18]. Algorithms especially designed for the PESP problem have also been developed, e. g., constraints propagation [24], genetic algorithms [19], branch-and-cut [15], constraint generation [20] or adapted backtracking algorithm [25]. These are specialized algorithms for finding feasible solutions quickly. However, for optimized solutions mostly ILP solvers are used.

2.2 Constraints

Various rules and restrictions that exist in the railway business can be modeled via PESP constraints of the form (1).

Trip time The trip time is the time needed for the train to run between two stations. Trip times do not necessarily need to be fixed, but can also be variable, as reported in [7]. The lower bound for the trip time is the minimum time needed for the train to run the distance plus a reserve of a few percent

that helps making the schedule more robust. The upper bound is the maximum acceptable time with respect to passenger patience and track capacity usage. The trip time (l, u) is a constraint between the departure and arrival events of the same train.

Dwell time The dwell time is the duration that a train stops in a station. This constraint connects arrival and departure event of a train. Dwell times should be long enough for boarding of new passengers and possibly for some loading/unloading or maintenance work on the train. It should not be much longer than necessary, however, as travelers would like to move on and platform capacity within a station might be small.

Connections These constraints relate the arrival event of some train to the departure event of another one in order to enable passengers to change trains. The minimum connection time depends on the infrastructure of the railway station, on the distances passengers have to walk. Upper bounds are again the acceptable waiting times for the travelers.

Headway The headway constraints are used to avoid collisions. They separate two trains running on the same track by at least the headway time h. This is done by introducing constraints $(h, T - h)$ between the arrival and the departure events of the two trains. It guarantees that the departures and arrivals of the two trains on the same track have a safe temporal distance. The headway time is only a simplification of the real safety system used in the railway world. More precise safety restrictions should be taken into account during the micro scheduling.

The headway constraints do not prevent overtaking of trains during the run on the same track, which is, of course, impossible without a collision. The problem can be solved by using more restrictive constraints, see [7] for details. The idea is to increase the headway times such that an overtaking is impossible even for the largest possible trip time difference. For example, a fast train with trip time $(30, 35)$ and a slow train $(35, 42)$ have a maximum trip time difference μ of 12 minutes. With a headway of h, the fast train would need to make up h minutes to catch up with the slow train and again h to restore the necessary headway before arrival at the destination. In the case of $\mu < 2h$ collisions can be excluded. In the example, this would require a headway time $h > 6$. If this condition is not fulfilled automatically, it can be achieved by lowering the trip time difference μ or by increasing the headway time h. Increasing headway should be avoided as it reduces the track capacity and flexibility. A different approach to cope with this problem is presented in Section 2.5.

All the above constraint types are of the form (1) and fit into the PESP model. Another constraint type will be introduced in Section 2.5, leading to an extended model. There are many others constraints that should be considered in the timetable generation and can be modeled as PESP constraints [22, 10].

2.3 Objective function

There are two classes of algorithms for solving the PESP: one looking for any feasible solution and the other looking for a solution that is optimal with re-

spect to a certain quality criterion. Feasibility algorithms are often much faster, as they stop as soon as the first feasible solution is found. Optimized solutions give a measure of the quality of a schedule and guarantee that the output is a solution of maximum quality. This guaranteed optimality is an advantage of the computer-generated railway timetables compared to the human-made ones. A description of possible optimization goals can be found in [22, page 57-64]. Typical goals are minimization of the total passenger travel time, minimization of the required number of train units or maximization of some measure of robustness. The objective functions used in this work are related to the flexible event slot concept introduced in Section 3.

2.4 Cycle periodicity formulation

The *Cycle Periodicity Formulation* (CPF) is an adapted formulation of the PESP that provides an alternative ILP formulation which turned out to be much more efficient in practice [10, 19, 21, 22]. Instead of solving for the event time variables π_i, it solves for periodic tensions x_{ij}. The tensions are the time differences between the two related events $x_{ij} = \pi_j - \pi_i + Tp_{ij}$ and must obey the bounds $l_a \leq x_a \leq u_a$ for each constraint $a \in A$. Additionally, for a periodic tension to have a periodic potential π_i at each node, the sum of all tensions along a cycle must be equal to an integer multiple of T, hence

$$\sum_{a \in C^+} x_a - \sum_{a \in C^-} x_a = Tq_C, \tag{2}$$

where q_C is the integer number of period jumps along the cycle C. This becomes intuitive by looking at the back transformation from the CPF variables x_{ij} to the PESP variables π_i. Starting by fixing any π_0, one can compute the neighboring values π using the relation $\pi_j = \pi_i + x_{ij} \mod T$, in short $\pi_j = [\pi_i + x_{ij}]_T$. As the same values for a π_i must result for any path one can take from a π_0, the sum of the x_{ij} along a cycle has to be an integer multiple of T. We obtain the following Cycle Periodicity Formulation (CPF):

$$\begin{aligned}
\text{minimize} \quad & f_{\text{obj}}(x) && && (3)\\
\text{s. t.} \quad & l_a \leq x_a \leq u_a, && \forall a \in A && (4)\\
& \sum_{a \in C^+} x_a - \sum_{a \in C^-} x_a = Tq_C, && \forall C \in G && (5)\\
& a_C \leq q_C \leq b_C, && \forall C \in G && (6)\\
& x_a \in \mathbb{R}, && \forall a \in A && (7)\\
& q_C \in \mathbb{Z}, && \forall C \in G && (8)
\end{aligned}$$

Eq. (5) imposes constraints on each cycle in the graph. The number of cycles in a graph can be exponential in the number of nodes, but it can be shown that there exist *integral cycle bases* B [12, 13] with the property that each cycle

C in G is a linear combination with coefficients from $\{-1, 0, +1\}$ of the cycles in B. Peeters [22] showed that it is sufficient to fulfill (5) for all $C \in B$. An integral cycle basis B of a graph G can be constructed by building a spanning tree Γ of G. When taking one chord $a \in A/\Gamma$ together with Γ, a graph with exactly one cycle occurs. Adding one cycle per chord to the basis B gives an integral cycle basis of G. For a PESP graph with n nodes and m arcs, the basis contains $|B| = m - (n-1)$ cycles, as the spanning tree of G has $n-1$ arcs. The advantage of the CPF over the original PESP formulation is that the search space can be reduced considerably by using the cutting planes (6) for the cycles in B [20]. The cycle basis is chosen such that it contains cycles with maximally restrictive cutting planes. The number of integer options for a q_C is denoted by $w_C = b_C - a_C + 1$. That gives a number of integer value combination to check of $\prod_{C \in B} w_C$ and can be reduced significantly by using a good cycle basis.

A theoretical discussion of minimal cycle bases can be found in [10] and many cycle basis construction heuristics are in [22]. Here, we always use the CPF formulation with an integer cycle basis generated using the minimum spanning tree approach, which is simple and gives good results in many cases. When using ILP solvers, it is important to find a good formulation to reach good performance. For the present case it is reported that the CPF formulation with a good cycle basis is more powerful than the original PESP [22, 10].

2.5 Non-collision constraints

The relation between periodic ordering and the q_C of the cycle can be used as non-collision constraint. Non-colliding trains have $q_C = 0$ on the cycle consisting of the two trip time arcs and the two headway arcs (which must have the same direction, e.g., from train 1 to 2). This fact has been reported earlier [24, 22, 10] and can also be adapted for non-collision constraints between trains of reversed direction on single tracks ($q_C = 0$ for the cycle consisting of the two trip time arcs having opposite direction and the two headway arcs with the same direction). The condition $q_C = 0$ is a type of collision constraint that does not fit into the original PESP framework, as it is not a proper PESP constraint. However, it can easily be added to the ILP formulation of both original PESP and CPF form.

The non-collision constraints $q_C = 0$ fit directly into the CPF formulation by choosing $a_C = b_C = 0$ in (6). Ideally, these non-collision cycles are used for the cycle basis, as they have the smallest possible $w_C = 1$.

3 Flexibility in the PESP

We can couple the macroscopic timetabling problem with the microscopic local scheduling by solving the PESP and passing the train departure and arrival times to the station routing algorithms to check feasibility. In order to avoid tedious iterations between micro and macro level in case of infeasibility of the micro-level problem, we want to improve the chance of finding a feasible solution

by increasing the solution space in the micro level. We can reach this goal if the PESP timetable does not impose exact event times π_i but enables some freedom for choosing the event times π_i. We can add this flexibility for the events π_i by introducing lower and upper bounds $\underline{\pi_i}$ and $\overline{\pi_i}$ for π_i as new decision variables instead of the event times π_i. The choice of the $\pi_i \in (\underline{\pi_i}, \overline{\pi_i})$ shall be independent for each event, i. e., each value $\pi_j \in (\underline{\pi_j}, \overline{\pi_j})$ should be reachable from each value in $\pi_i \in (\underline{\pi_i}, \overline{\pi_i})$ by remaining feasible in the sense of Eq. (1). Note that we are not forced to add this flexibility to all the events, but we can select the nodes where we want to add it, for instance only nodes corresponding to events in a main station area with high traffic density, where it is more difficult to schedule trains on the microscopic level. The micro scheduling algorithm proposed in [6, 1] is designed to cope with such flexible event time inputs. Here we present a new approach how the PESP can be generalized to generate event slot timetables on a macroscopic level.

Flexible schedules with event time slots $\pi_i \in (\underline{\pi_i}, \overline{\pi_i})$ can also be used to overcome delay propagation in the network. A train has to leave a station at time $\overline{\pi_i}$ at the latest without starting a delay cascade on following event times. If the departure time π_i is scheduled earlier than $\overline{\pi_i}$, the difference $\overline{\pi_i} - \pi_i$ can be used to compensate delays and make the schedule robust. The local train routing algorithm should therefore preferably choose event times $\pi_i \in (\underline{\pi_i}, \overline{\pi_i})$ that are close to $\underline{\pi_i}$ such that the remaining flexibility $\overline{\pi_i} - \pi_i$ can be maximized. Related interesting approaches to impose robustness against delays in the PESP environment can be found in [8, 14]. In particular, [14] introduces the notion of absorbing path, which is a path that absorbes a limited disturbance occured at the first arc at least by the end of the path. This is achieved by adding time reserves to the lower bounds of the PESP formulation. Our approach also restricts the feasible intervals on the arcs, but instead of associating these new variables directly with the arcs, we decide to associate them with the events of the network. Doing so, these variables serve as a measure for their events' flexibility for microscopic scheduling and might lead to additional robustness on the operational level.

3.1 Basic properties

We set the ranges for the event time bounds as $0 \leq \underline{\pi_i} < T$ for the lower bound and $\underline{\pi_i} \leq \overline{\pi_i} < \underline{\pi_i} + T$ for the upper bound. Thus, we define the flexibility δ_i of an event i as the size of its time slot

$$\delta_i := \overline{\pi_i} - \underline{\pi_i}. \tag{9}$$

Each constraint arc $(i, j) \in A$ has a correspondent span $\gamma_{ij} = u_{ij} - l_{ij}$. From an event $\pi_i \in (\underline{\pi_i}, \overline{\pi_i})$ with flexibility δ_i, another event $\pi_j \in (\underline{\pi_j}, \overline{\pi_j})$ must be reachable by fulfilling the constraint $l_{ij} \leq \pi_j - \pi_i + p_{ij}T \leq u_{ij}$, as illustrated in Figure 1. When no other constraints apply to π_j then $\underline{\pi_j} = [\overline{\pi_i} + l_{ij}]_T$ is the first possible time for event j that fulfills constraint (i, j) for any $\pi_i \in (\underline{\pi_i}, \overline{\pi_i})$.

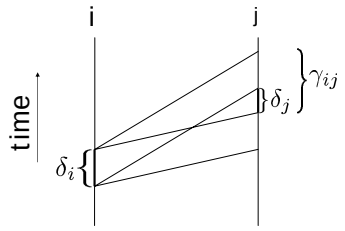


Fig. 1. Flexibility for the events i and j. By increasing the flexibility δ_i, the flexibility for the connected node will be reduced by the same amount such that the sum of both values is at most the arc span γ_{ij}.

Similarly, $\overline{\pi_j} = [\underline{\pi_i} + u_{ij}]_T$. The flexibility δ_j is then given by $\delta_j = \overline{\pi_j} - \underline{\pi_j} = [(\underline{\pi_i} + u_{ij}) - (\overline{\pi_i} + l_{ij})]_T = \gamma_{ij} - \delta_i$. It follows that

$$\delta_i + \delta_j \leq \gamma_{ij} \tag{10}$$

This inequality takes into account that other constraints besides (i, j) could restrict the flexibility of the nodes further. Thus, each node flexibility of a feasible timetable is a non-negative value $\delta_i \geq 0$. Note that finding a set of non-negative δ_i fulfilling (10) does not guarantee a feasible timetable. For instance, if we choose all $\delta = 0$ in an infeasible PESP instance we satisfy Eq. (10) but the problem is infeasible. Eq. (10) shows that the event flexibilities are dependent. Adding more flexibility at one node restricts it at the neighbors. A weighted objective function or a feedback strategy from the microscopic algorithm could then help allocate flexibility where it is most useful.

3.2 Flexible PESP model

We now present the model for introducing event flexibility into the PESP by changing the constraints of the PESP graph. Event time slots require that the PESP constraints are fulfilled for any $\pi_i \in (\underline{\pi_i}, \overline{\pi_i})$, independently for each event. The range of the time span $x_{ij} = \pi_j - \pi_i + Tp_{ij}$ between two events i and j is given by

$$\underline{\pi_j} - \overline{\pi_i} + Tp_{ij} \leq \pi_j - \pi_i + Tp_{ij} \leq \overline{\pi_j} - \underline{\pi_i} + Tp_{ij} \tag{11}$$

Replacing the upper bounds $\overline{\pi_i}$ for the event times with $\underline{\pi_i} + \delta_i$ we get the following inequalities:

$$\underline{\pi_j} - (\underline{\pi_i} + \delta_i) + Tp_{ij} \leq \pi_j - \pi_i + Tp_{ij} \leq (\underline{\pi_j} + \delta_j) - \underline{\pi_i} + Tp_{ij}. \tag{12}$$

The PESP constraints (1) are satisfied for any combination of π_i and π_j if they are satisfied for the entire range of $(\pi_j - \pi_i + Tp_{ij})$ in Eq. (12):

$$l_{ij} \leq \underline{\pi_j} - (\underline{\pi_i} + \delta_i) + Tp_{ij} \leq \pi_j - \pi_i + Tp_{ij} \leq (\underline{\pi_j} + \delta_j) - \underline{\pi_i} + Tp_{ij} \leq u_{ij}. \tag{13}$$

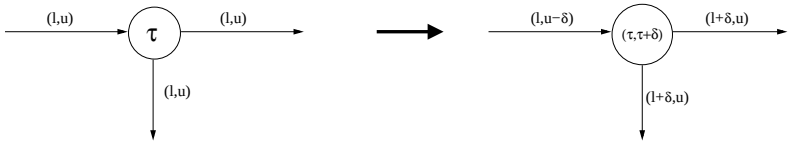


Fig. 2. Introducing an event slot of size δ_i at event i leads to adapted constraint bounds in the PESP graph. The upper bound of incoming constraints is reduced by δ_i and the lower bound of outgoing arcs is increased by δ_i.

Considering separately the first and the last inequalities we obtain constraints in PESP form for the variables $\underline{\pi_i}$.

$$l_{ij} + \delta_i \leq \underline{\pi_j} - \underline{\pi_i} + Tp_{ij} \quad \text{and} \quad \underline{\pi_j} - \underline{\pi_i} + Tp_{ij} \leq u_{ij} - \delta_j. \tag{14}$$

Putting these results together leads to

$$l_{ij} + \delta_i \leq \underline{\pi_j} - \underline{\pi_i} + Tp_{ij} \leq u_{ij} - \delta_j \tag{15}$$

which are constraints in PESP form for the variables $\underline{\pi_i}$. The adaptation of the constraints is illustrated in Figure 2. The constraints are more restrictive than in the original PESP, $\tilde{\gamma_{ij}} = (u_{ij} - \delta_j) - (l_{ij} + \delta_i) = \gamma_{ij} - \delta_i - \delta_j$. As $\tilde{\gamma_{ij}}$ must be non-negative for feasibility, it follows again $\delta_i - \delta_j \leq \gamma_{ij}$ as stated before in Eq. (10). The original PESP without event slots is the special case where $\delta_i = 0$ for all i.

The Flexible PESP can now be solved for the variables $\underline{\pi}$ and δ. Both the original PESP and the CPF formulation are applicable. In the original PESP, Eq. (1) changes to

$$l_{ij} + \delta_i \leq \underline{\pi_j} - \underline{\pi_i} + Tp_{ij} \leq u_{ij} - \delta_j \quad \forall\ (i,j). \tag{16}$$

In the CPF version, the change affects the bounds of Eq. (4) as follows:

$$l_{ij} + \delta_i \leq x_{ij} \leq u_{ij} - \delta_j \quad \forall\ (i,j). \tag{17}$$

3.3 Objective functions

A good timetable with time slots should (i) be a good timetable with respect to the objectives in Section 2.3, (ii) have large event time slots, and (iii) contain homogeneously distributed event flexibility. These goals are often conflicting, and the choice of the objective function is not obvious. The following list discusses some possible choices. Computational results for different objectives are presented later in Section 4.3.

– MINTRAVEL: This objective function minimizes a weighted sum of the passenger-relevant times

$$\min f_{\text{tt}} = \sum_{t \in A_T} w_t x_t + \sum_{d \in A_D} w_d x_d + \sum_{c \in A_C} w_c x_c. \tag{18}$$

where $A_T \subseteq A$ is the set of trip arcs, $A_D \subseteq A$ the set of dwell arcs and $A_C \subseteq A$ the set of connection arcs. The weights can be chosen such that they correspond to the number of passengers using this activity or other priority criteria.

- MAXFLEX: This objective function maximizes a weighted sum of flexibilities

$$\max f_{\text{flex}} = \sum_{i \in V} w_i \delta_i \tag{19}$$

where V is the set of all events where flexibility is introduced. The weights can be chosen such that more flexibility is assigned to some parts of the graph, e.g., main station areas or network bottlenecks. This objective (19) may lead to a few events having a lot of flexibility while all others have none. It is more desirable to have a bit of flexibility everywhere. By additionally constraining the maximum flexibility per node,

$$\delta_i \leq \delta_{max} \tag{20}$$

a better distribution of flexibility can be obtained. Different choices for the value δ_{max} are discussed in Section 4.

- MIXFLEX: An aggregated objective function allows to optimize both the quality of the timetable and the time slots. The timetable quality here is measured by a weighted sum, whose optimum constitutes a Pareto-optimal solution to the bi-objective problem of minimizing travel time and maximizing flexibility simultaneously. The weight λ, $0 < \lambda < 1$, balances the priorities of the two goals.

$$\max f_{\text{mixflex}} = \lambda \cdot f_{flex} - (1 - \lambda) \cdot f_{tt} \tag{21}$$

- CONTRAVEL: Instead of optimizing a weighted sum of the objectives, we can address the bi-objective problem by constraining one objective and optimizing the other. By appropriate constraint values, any Pareto-optimal solution is reachable, and the quality of the final solution can be controlled more accurately than via a weighted sum. Here, we optimize flexibility under a travel time constraint, where we use the minimum of f_{tt} as a reference and allow a parameterized relative deviation of ϵ:

$$\max f_{\text{flex}} \tag{22}$$

$$\text{subject to} \quad f_{\text{tt}} \leq (1 + \epsilon)\ f^*_{\text{tt}} \tag{23}$$

where f^*_{tt} is the optimal value found for f_{tt} in (18).

- POSTOPT: Another two step approach keeps the integer variables q_C from step one (18) fixed for step two (22). It results an LP, all integer variables now being fixed as $q_C = q^*_C$. This is a type of post optimization, which is very fast, but has a very limited solution space. The second step only shifts the event times while keeping the event order constant. A similar post-optimization approach has been applied in [8] for finding an optimal distribution of time reserves among a train trip using stochastic optimization.

– MAXMINFLEX: The idea here is to guarantee a minimum flexibility for a set of selected events $i \in \Psi$

$$\max \varphi \tag{24}$$

$$s.t.\ \varphi \leq \delta_i \quad \forall i \in \Psi. \tag{25}$$

In many cases, however, there are events that cannot have any flexibility. In such a case, φ will be zero and the approach will not give the desired results.

3.4 Interaction with micro-level scheduling

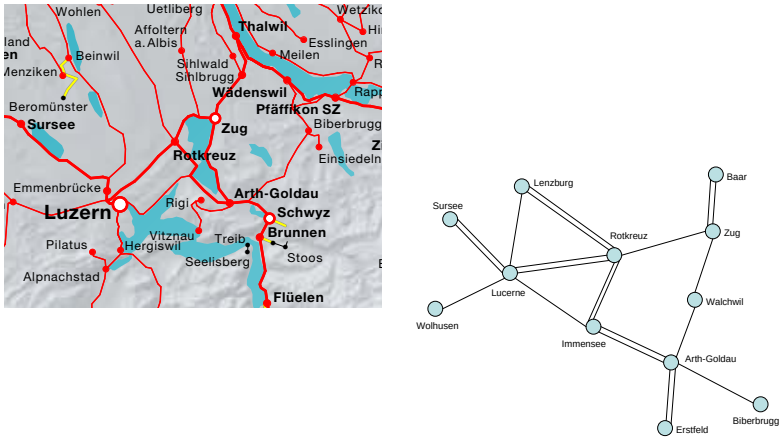
When the optimal macro schedule is found, the event times and event slot sizes are passed to the micro scheduling algorithm, where routes and platforms are assigned to the trains. The event slots increase the solution space of the micro scheduling, which can now choose from various routing possibilities for each train, as well as from the event times $\pi \in (\underline{\pi}, \underline{\pi} + \delta)$ within the slots. The interface consists of a list of trains and their π and δ values for the arrival and departure at each station. If no solution of the micro scheduling problem can be found, a feedback loop leads to a shift of the weights, w_i in (19) and λ in (21), in the objective function of the macro scheduling. More flexibility is then assigned to the respective station area.

4 Computational results

4.1 Test case

A test case was set up in order to validate the algorithms and concepts of this work. The scenario includes the cities Lucerne, Zug and Arth-Goldau as the major nodes in the network. The macroscopic track topology shown in Figure 3 is used for the test case. It is a simplification of the reality, but it is still interesting, as it includes changes from double to single track and junctions where trains from different directions come together, as well as a mixture of freight trains, long distance and local passenger trains.

The service intention of the 2007 SBB timetable is used. It contains Intercity trains running from Baar (Zurich) and Sursee (Basel) to Erstfeld and the Gotthard tunnel through the Alps. Additionally, there are Interregional trains running from Lucerne to Baar and to Biberbrugg (Pfäffikon). Regional trains run in the triangle Lucerne – Zug – Arth-Goldau with several stops in between, as well as on all other lines described in Figure 3 (b). Several slots for cargo trains are reserved in every hour on the double-track line Lenzburg – Rotkreuz – Immensee – Arth-Goldau – Gotthard in both directions, which is the main line between Germany and Italy and where nearly the entire freight traffic passes through. The reference scenario for the test case consists of 48 trains running on the described topology with a periodicity of 1 hour ($T = 60$ min) and the headway time $h = 2$ min. Table 1 compares sizes of the PESP graph and the MIP formulation for the reference scenario with and without flexibility.



(a) (b)

Fig. 3. (a) The test case region connecting the towns Zug – Lucerne – Arth Goldau in central Switzerland. (b) The track topology is partly double track and partly single track and is used by regional and intercity trains as well as freight trains. All the events in the PESP model correspond to departure or arrival times at stations in this picture.

# variables	# integer variables	# δ variables	# MIP constraints	# non-collision	average (stdev) arc span (min)	average ω_C (stdev)
1083	436	212	1730	223	5.4 (4.8)	2.6 (1.1)

Table 1. Data of the PESP graph and the MIP for the reference scenario, with the variables δ for adding flexibility. The PESP graph of the reference scenario with 48 trains has 212 events and 647 arcs after resolving arcs with zero span. The average arc span and his standard deviation are computed without the headway constraints, wich are 446 arcs with span 56 minutes ($h = 2$).

4.2 Implementation

The model was implemented using Matlab® and the MOSEK® [17] solver for mixed integer linear programs. The tests are run on a 2GHz 64bit processor with 4GB RAM. All computations throughout this chapter are done with all weights w_i equal to one and are terminated when an optimality gap of 0.1% is reached. The output of the timetable generator is a list of all departure and arrival times of the trains. The data can then be visualized in the form of time-space diagrams (see Figure 4).

The timetable can be computed with both the original PESP formulation (Section 2.1) or the CPF formulation (Section 2.4). First we compare the two formulations as well as the model with and without flexibility on the events. We consider the objective function MINTRAVEL. The reference scenario takes more than 4000 seconds when we apply the original PESP formulation but only 18 s when we apply the CPF formulation. As often observed in the literature, it

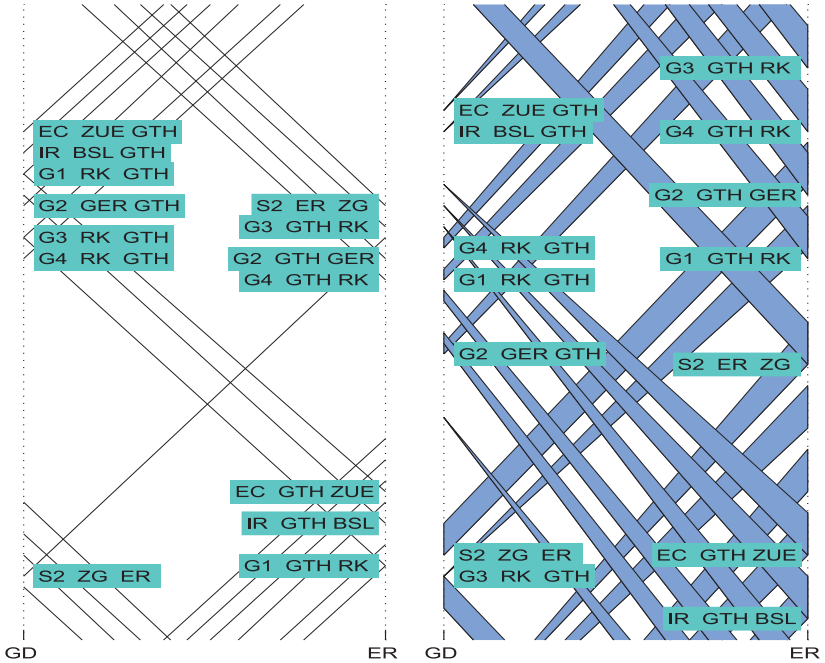


Fig. 4. (a) The generated timetable without event slots from the 2007 SBB service intention visualized in a time-space diagram. The horizontal axis represents the route from Arth-Goldau (GD) towards the alps (Erstfeld, ER), whereas the vertical axis represents the time. (b) When using event slots, each event gets an event time $\underline{\pi_i}$ and a flexibility δ_i. In this diagram, the earliest possible line and the latest possible line are filled in grey. Any choice for train trajectories in the grey zones are feasible from the macro scheduling point of view.

seems that the CPF formulation is more efficient and better suited for timetable generation. It is therefore used for the further tests throughout this section.

If we introduce the values δ for the event flexibility, consider the CPF formulation and solve the reference scenario with the objective MINTRAVEL, we get a CPU time of 275 seconds, as reported in Table 2. Other tested scenarios give similar increasing factors of the CPU time by introducing event flexibility. Notice that an optimal solution of MINTRAVEL with all $\delta_i = 0$ corresponds to optimal solution of the original PESP without flexibility. It is interesting to notice that the MIP solver of MOSEK does not find this solution, but takes more time and provides a solution with $\sum \delta_i = 60$. If we solve CONTRAVEL with $\epsilon = 0$ we get an optimal solution with $\sum \delta_i = 152$.

If we want to maximize the flexibility in the objective (MAXFLEX), we get a CPU time of 420 seconds. One can observe that the introduction of the additional values δ, which more or less doubles the number of continuous variables in the ILP, increases the CPU time, but not too much as we did not add any additional

name	objective	CPU time	$\sum \delta_i$	$\sum x_{t,d,c}$
NOFLEX	$\min \sum x_{t,d,c}$	18 sec	-	2017
MINTRAVEL	$\min \sum x_{t,d,c}$	210 sec	67	2017
MAXFLEX	$\max \sum \delta_i$	420 sec	380	2396
MIXFLEX 1/2	$\max \sum \delta_i - \sum x_{t,d,c}$	217 sec	249	2114
MIXFLEX 2/3	$\max 2 \cdot \sum \delta_i - \sum x_{t,d,c}$	338 sec	372	2251
MIXFLEX 9/10	$\max 9 \cdot \sum \delta_i - \sum x_{t,d,c}$	317 sec	380	2272
POSTOPT	$\max \sum \delta_i$ for fixed q_C	0.2 sec	251	2121
CONTRAVEL	$\max \sum \delta_i$ s. t. $\sum x_{t,d,c} \leq 1.02 \cdot f^*_{\text{tt}}$	93 sec	194	2058

Table 2. Results for the reference scenario with bounds for the flexibilities $\delta_i \leq 4$. $\sum x_{t,d,c}$ stands for the sum of all trip, dwell and connection times. NOFLEX means the original PESP solved with CPF formulation, without introducing the variables δ. Notice that for POSTOPT and CONTRAVEL a solution to NOFLEX is needed; the reported CPU time is without the time needed for NOFLEX.

integer variables. Furthermore, an appropriate choice of the objective could help to improve the CPU time (see Table 2). Results on the test case with event slots are displayed in Figure 4. Here, the reference scenario with 48 trains is used, with a limitation of the event slot sizes to $\delta_i \leq 4$.

4.3 Event slot objectives

The limitation of the event flexibilities $\delta_i \leq \delta_{max}$ has several reasons. Large flexibilities are not very useful, neither for the micro scheduling nor for the delay management. On the contrary, events with large δ_i restrict the δ_j for other events because $\delta_j \leq \gamma_{ij} - \delta_i$. It is better to have many small time slots instead of a few large ones. Table 3 shows the effect of the flexibility bounds.

A second drawback of large flexibilities is that the travel times are necessarily increased, as the minimal bound for the trip times is increased in Eq. (15). This is only acceptable if the increase is small and if it is compensated by additional timetable robustness.

Generating a timetable with maximized flexibility needs more computation time. The increase can be explained by comparing the effects of the objective functions on the solver. An objective function that minimizes the trip and connection times (MINTRAVEL, see Table 2) automatically saves the capacity of the tracks by trying to assign to each train the shortest track occupancy time possible. The objective function basically helps the solver to find a solution, as it is more probable to find one when the trains use only little track capacity. MINTRAVEL has the additional advantage of offering passenger-friendly train schedules with low travel times.

An objective function maximizing the event slots (MAXFLEX) has the inverse effect. An event with high flexibility also uses a lot of track capacity. This can be seen in Figure 4, where the flexible events occupy a band (filled in grey) instead of just a single line. With such an objective function the solver starts looking for

δ_{max}	$\sum \delta_i$	number of events with				
		$\delta_i = 0$	$\delta_i = 1$	$\delta_i = 2$	$\delta_i = 3$	$\delta_i \geq 4$
0	0	212	–	–	–	–
1	161	51	161	–	–	–
2	258	61	44	107	–	–
3	323	67	48	16	81	–
4	366	76	42	16	20	58
5	386	79	44	17	14	58
6	396	80	49	15	14	54
7	401	84	44	21	13	50
8	405	82	52	16	13	49
9	409	94	43	12	10	53
10	411	95	42	12	10	53
11	413	94	40	16	10	52
12	415	92	44	14	14	48
59	419	97	42	13	10	50

Table 3. This table shows the effect of the limitation $\delta_i \leq \delta_{max}$ when MAXFLEX is optimized in the reference scenario. The choice of the δ_{max} has the conflicting goal of maximizing $\sum \delta_i$ while minimizing the number of events with zero flexibility. For the following tests, the flexibility bound $\delta_{max} = 4$ is used.

solutions with high flexibility, which are not likely to be feasible as they block a lot of track capacity. Hence the approaches which combine the advantages of both are desirable. MIXFLEX $\frac{2}{3}$ (see Table 2) is an aggregated objective function giving good values for trip times and flexibility within a reasonable time.

The post optimization approach (POSTOPT) takes the quickly generated MINTRAVEL solution and adds flexibility in a second step while keeping the integer variables q_C constant. The ILP is reduced to an LP for the second step and is therefore solved almost instantaneously. It is interesting to see that the resulting flexibility is quite high, even compared to the maximally possible objective in MAXFLEX. It can be expected that the computation times of MAXFLEX grow faster than MINTRAVEL with the problem size due to the capacity problem. This makes the POSTOPT concept attractive for larger instances.

The CONTRAVEL works on a reduced search space that contains only the solutions with the given maximum deviation to the optimum of MINTRAVEL. It is interesting to see that the computation time of this approach depends much on the instances. For the reference scenario it provided good results but for some other instances it did not come to an optimality proof after more than ten hours. The reason might be that the travel time restriction does not give a reduction of the search space of the integer variables.

5 Conclusions and outlook

The classical PESP model with fixed event times is quite restrictive and could lead to a draft timetable which is infeasible at the microscopic level. It is therefore

desirable to increase the solution space for the microscopic level by enabling the event times to be in a time slot instead of being fixed to an exact value. This paper shows how this idea can be modeled by generalizing the PESP for generating flexible train schedules on a macroscopic level. The resulting FPESP is closely related to the original PESP such that future improvements in the area can probably be included.

Computations show that we can generate draft timetables on the macroscopic level with event slots flexibility for a scenario of medium size (48 trains in one hour) in a reasonable amount of time (2-7 minutes). The introduction of the event slots does not seem to affect the computation time too much and should be compensated by the reduction of the number of iterations between the macro and micro level.

An important result of the event slot tests was that the computation time strongly depends on the objective function. The event slot maximization is computationally not very efficient and conflicts with the goal of travel time minimization. We have tested some alternative objective functions and observed that the problem can partly be overcome with aggregated objective functions.

The integration of the macro and micro level is currently under investigation. Draft timetables generated with the approach presented in this paper have to be checked for feasibility at the micro level with algorithms designed to cope with this type of event slot timetables. Of particular interest is the measure of the increased chance of avoiding an infeasible instance and therefore the restart of the timetable generation on the macro level.

Moreover, larger scenarios should be tested with the model, such for instance a larger area or the complete Swiss Intercity network. Larger scenarios will help understand the limits of this model from a computational time point of view, in particular to see whether it allows to generate schedules for the whole Swiss railway network.

References

1. G. Caimi, F. Chudak, M. Fuchsberger, and M. Laumanns. Solving the train scheduling problem in a main station area via a resource constrained space-time integer multi-commodity flow. Technical report, Institute for Operations Research, ETH Zurich, 2007.
2. G. Caimi, T. Herrmann, D. Burkolter, F. Chudak, and M. Laumanns. Design of a new railway scheduling model for dense services. In *Proceedings of the 2nd International Seminar on Railway Operations Modelling and Analysis (RailHannover 2007), Hannover, Germany*. IAROR, 2007.
3. A. Caprara, M. Fischetti, and P. Toth. Modeling and solving the train timetabling problem. *Operations Research*, 50(5):851–861, 2002.
4. M. Carey. A Model and Strategy for Train Pathing with choice of lines, platforms, and routes. *Transportation Research Part B*, 28(5):333–353, 1994.
5. M. Ehrgott, R. Velasquez, and A. Schöbel. A Set-packing Approach to Routing Trains Through Railway Station. Preprint nr. 2005-36, Georg August Universität Göttingen, 2005.

6. M. Fuchsberger. Solving the train scheduling problem in a main station area via a resource constrained space-time integer multi-commodity flow. Master's thesis, ETH Zurich, 2007.
7. L. Kroon and L. Peeters. A Variable Trip Time Model for Cyclic Railway Timetabling. *Transportation Science*, 37(2):198–212, 2003.
8. L. G. Kroon, R. Dekker, and M.J.C.M. Vromans. Cyclic railway timetabling: a stochastic optimization approach. Technical report, RSM Erasmus University, 2005. available at http://hdl.handle.net/1765/6957.
9. F. Laube, S. Roos, R. Wüst, M. Lüthi, and U. Weidmann. PULS 90 - ein systemumfassender Ansatz zur Leistungssteigerung von Eisenbahnnetzen. *Eisenbahntechnische Rundschau*, 3/2007, 2007. In German.
10. C. Liebchen. *Periodic Timetable Optimization in Public Transport.* PhD thesis, Technische Universität Berlin, 2006.
11. C. Liebchen and R. Möhring. The Modeling Power of the Periodic Event Scheduling Problem: Railway Timetables - and Beyond. In F. Geraets et al., editors, *ATMOS 2004*, LNCS 4359. Springer, 2004.
12. C. Liebchen and L. Peeters. On cyclic timetabling and cycles in graphs. Technical Report 761-2002, TU Berlin, Department of Mathematics, Combinatorial Optimization and Graph Algorithms Group, 2002.
13. C. Liebchen and R. Rizzi. Classes of cycle bases. *Discrete Applied Mathematics*, 155(3):337–355, 2007.
14. C. Liebchen and S. Stiller. Delay resistant timetabling. Technical Report 24-2006, TU Berlin, Department of Mathematics, Combinatorial Optimization and Graph Algorithms Group, 2006.
15. T. Lindner. *Train Schedule Optimization in Public Rail Transport.* PhD thesis, Technische Universität Braunschweig, June 2000.
16. M. Lüthi, A. Nash, U. Weidmann, F. Laube, and R. Wüst. Increasing railway capacity and reliability through integrated real-time rescheduling. In *Proceedings of the 11th World Conference on Transport Research, Berkeley*, 2007.
17. MOSEK ApS, Copenhagen, Denmark. *The MOSEK optimization manual*, 2007. Version 5.0.0.57, Available at www.mosek.com.
18. K. Nachtigall. Periodic Network Optimization and Fixed Interval Timetables. Habilitation Thesis, University Hildesheim, 1998.
19. K. Nachtigall and S. Voget. A genetic algorithm approach to periodic railway synchronization. *Computers & OR*, 23(5):453–463, 1996.
20. M.A. Odijk. A constraint generation algorithm for the construction of periodic railway timetables. *Transportation Research Part B*, 30(6):455–464, 1996.
21. L. Peeters and L. Kroon. A cycle based optimization model for the cyclic railway timetabling problem. In S. Voß and J.R. Daduna, editors, *Proceedings Computer-Aided Scheduling of Public Transport (CASPT 2000)*, volume 505, pages 275–296. Springer, Berlin, 2001.
22. L.W.P. Peeters. *Cyclic Railway Timetable Optimization.* PhD thesis, Erasmus University Rotterdam, 2003.
23. G. Sahin, R. K. Ahuja, and C. B. Cunha. New approaches for the train dispatching problem. *submitted to Transportation Research Part B*, 2004.
24. A. Schrijver and A. Steenbeck. Dienstregelingontwikkeling voor railned (timetable construction for Railned). Technical report, C.W.I. Center for Mathematics and Computer Science, Amsterdam, 1994. In Dutch.
25. P. Serafini and W. Ukovich. A mathematical model for periodic scheduling problems. *SIAM J. Disc. Math.*, 2(4):550–581, 1989.

26. R. Wüst. Dynamic rescheduling based on predefined track slots. In *Proceedings of 7th World Congress on Railway Research, Montreal*, 2006.
27. P. J. Zwaneveld, L. G. Kroon, H. E. Romeijn, M. Salomon, S. Dauzère-Pérès, S. P. M. Van Hoesel, and H. W. Ambergen. Routing Trains through Railway Stations: Model Formulation and Algorithms. *Transportation Science*, 30(3):181–194, August 1996.