Forward Cycle Bases and Periodic Timetabling

Niels Lindner[1] ✉ 🆔
Zuse Institute Berlin, Germany

Christian Liebchen ✉ 🆔
Technical University of Applied Sciences Wildau, Germany

Berenike Masing ✉ 🆔
Zuse Institute Berlin, Germany

---- **Abstract** ----

Periodic timetable optimization problems in public transport can be modeled as mixed-integer linear programs by means of the Periodic Event Scheduling Problem (PESP). In order to keep the branch-and-bound tree small, minimum integral cycle bases have been proven successful. We examine forward cycle bases, where no cycle is allowed to contain a backward arc. After reviewing the theory of these bases, we describe the construction of an integral forward cycle basis on a line-based event-activity network. Adding turnarounds to the instance `R1L1` of the benchmark library PESPlib, we computationally evaluate three types of forward cycle bases in the Pareto sense, and come up with significant improvements concerning dual bounds.

1 Introduction

As any mathematical optimization problem, periodic timetable optimization is about detecting a feasible solution with the best possible objective value. To solve the mixed integer programs that arise from the *Periodic Event Scheduling Problem* (PESP) [25], a standard model for periodic timetable optimization, to optimality, it is of interest to focus on those parts of the feasible region that allow for good objective values in particular. Hence, quite an amount of research has been dedicated to identifying valid inequalities for the PESP polytope, and making use of them as cutting planes in a branch-and-cut context [2, 15, 16, 19, 20, 21, 23]. Following the general goal of describing the convex hull of the integer points as sharp as possible, these approaches focus on the entire feasible region. Yet, as we deal with an optimization problem, it is of particular interest to examine those parts of the feasible region that allow for good objective values.

Most of the known successful cutting planes for PESP can be derived from oriented cycles in the underlying event-activity network. However, as these cycles contain in general both forward and backward arcs, it is hard to establish a link between constraints from cycles, typically containing coefficients with different signs, and the objective function, whose coefficients are all non-negative.

---

[1] corresponding author

We therefore propose to focus on *forward cycles*, i.e., oriented cycles that contain exclusively forward arcs. Then the classical cycle inequalities by Odijk [23] have entirely non-negative coefficients for the travel time variables, so that tight constraints directly relate to the objective function.

In the area of periodic timetabling, there is one collection of reference instances publicly available, the so-called PESPlib [6]. To date, for all of its instances their best known solutions still show relatively large duality gaps. Having a closer look to the railway-motivated instances `RxLy`, providing some feasible solution is trivial, because when removing all free arcs, i.e., arcs for which any travel time modulo the period time is feasible, these instances essentially decompose into paths [7, 17]. Hence, the main challenge for the `RxLy` instances of the PESPlib is about the actual optimization process rather than feasibility. We want to support this process better by seeking in particular those valid inequalities which have a significant impact on the objective function.

We propose to modify the instances `RxLy` slightly: We match the paths mentioned above to lines, and add turnaround arcs at the line ends. It then becomes possible to find a cycle basis consisting of forward cycles only. Moreover, turnarounds allow for an investigation of vehicle rotations, so that we can discuss optimality in the Pareto sense with respect to both the minimum passenger travel time and the minimum number of vehicles. Varying minimum turnaround times and turnaround weights, it turns out that the forward cycles approach is fruitful, and we can compute a new dual incumbent for the instance `R1L1`.

We formally describe the Periodic Event Scheduling Problem in Section 2. In Section 3, we motivate the use of forward cycles in detail, review the theory of forward cycles bases, and provide a construction for an integral cycle basis for timetabling networks with a specific structure. Analyzing the PESPlib benchmark instance `R1L1`, we describe in Section 4 how to add turnaround arcs in a meaningful way, which allows to compute forward cycle bases on a modified instance `R1L1v`. Section 5 presents computational results, evaluating passenger travel time slack and the number of required vehicles for different choices of cycle bases, minimum turnaround times, and turnaround weights. We conclude the paper in Section 6.

2 Periodic Event Scheduling

2.1 Problem Definition

The *Periodic Event Scheduling Problem* (PESP) has been introduced by Serafini and Ukovich [25]. A PESP instance consists of a 5-tuple (G, T, ℓ, u, w), where

- $G = (V, A)$ is a directed graph, often called *event-activity network*,
- $T \in \mathbb{N}$ is a *period time*,
- $\ell \in \mathbb{R}_{\geq 0}^A$ is a vector of *lower bounds* with $0 \leq \ell < T$,
- $u \in \mathbb{R}_{\geq 0}^A$ is a vector of *upper bounds* such that $0 \leq u - \ell < T$,
- $w \in \mathbb{R}_{\geq 0}^A$ is a vector of *weights*.

For an instance (G, T, ℓ, u, w), a vector $\pi \in [0, T)^V$ is called a *periodic timetable* if there is a *periodic tension* $x \in \mathbb{R}^A$ such that

$$\forall ij \in A : \quad \ell_{ij} \leq x_{ij} \leq u_{ij} \quad \text{and} \quad x_{ij} \equiv \pi_j - \pi_i \mod T.$$

If x is a periodic tension, then $y := x - \ell$ is called *periodic slack*.

Interpreting the vertices in V as events and the arcs in A as activities, a periodic timetable π fixes event timings, and a periodic tension x is an assignment of activity durations, both modulo the period time T. In the context of periodic timetabling, events are typically arrivals or departures of vehicles at stations, and activities model driving between stations, dwelling or transferring at a station, turnarounds, etc. [13].

▶ **Definition 1** ([25]). *Given (G, T, ℓ, u, w), the* Periodic Event Scheduling Problem *(PESP) is to find a periodic timetable π along with a periodic tension x such that the weighted slack $\sum_{a \in A} w_a y_a$ is minimum or to decide that no periodic timetable exists.*

As the arc weights w often reflect the number of passengers using a specific activity, and periodic tensions correspond to activity durations, the PESP objective then amounts to minimize the total passenger travel time. It is also possible to shift the focus onto the number of vehicles by introducing large weights on turnaround activities [13]. We will evaluate both weighted slack and number of vehicles in our computational Section 5.

2.2   Cycle-Based Mixed-Integer Program

We will use the following cycle-based mixed-integer programming formulation for PESP throughout this paper:

$$
\begin{aligned}
\text{Minimize} \quad & w^\top y \\
\text{s.t.} \quad & \Gamma(y + \ell) = Tz, \\
& 0 \le y \le u - \ell, \\
& z \in \mathbb{Z}^B.
\end{aligned}
\tag{1}
$$

In (1), B is an integral cycle basis of G with cycle matrix Γ. We refer to Section 3.2 or [12] for details. The integer z-variables model the modulo T conditions, we will call them *cycle offset variables*. Periodic timetables are only implicit in this formulation, for a feasible periodic slack y, a timetable can be recovered by a graph traversal.

We conclude this section by recalling a class of valid inequalities. An *oriented cycle* in G is a vector $\gamma \in \{-1, 0, 1\}^A$ such that $\{a \in A \mid \gamma_a \neq 0\}$ constitutes a cycle in the undirected graph arising from G by forgetting the arc orientations. We can decompose $\gamma = \gamma_+ - \gamma_-$ into its positive (forward) and negative (backward) part $\gamma_+ \in \{0, 1\}^A$ and $\gamma_- \in \{0, 1\}^A$, respectively.

▶ **Theorem 2** ([23]). *Let (G, T, ℓ, u, w) be a PESP instance. Let $\gamma \in \{-1, 0, 1\}^A$ be an oriented cycle in G. Then the* cycle inequalities

$$
\left\lceil \frac{\gamma_+^\top \ell - \gamma_-^\top u}{T} \right\rceil \le \frac{\gamma^\top(y + \ell)}{T} \le \left\lfloor \frac{\gamma_+^\top u - \gamma_-^\top \ell}{T} \right\rfloor
\tag{2}
$$

are valid for all feasible periodic slacks y.

Since the rows of Γ are composed of oriented cycles, the cycle inequalities (2) may also be used to determine bounds on the cycle offset variables z in (1).

3   Forward Cycle Bases

3.1   Motivation

Let us first highlight a few observations about the mixed-integer program (1) and the cycle inequalities (2):

- The overall objective function in (1) is composed of the objective values that arise on the individual arcs.
- The contribution of one individual arc a to the objective function is a monotonic increasing function of the slack y_a.
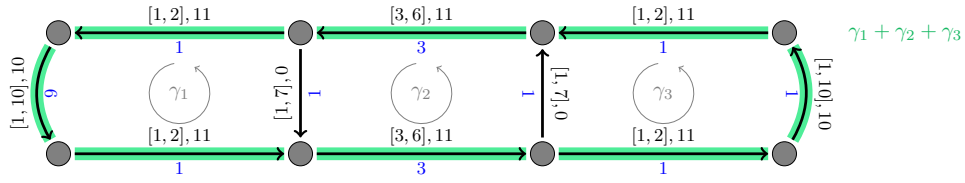
Figure 1 Example event-activity network with period time $T = 10$ and labels $\xrightarrow{\;[\ell_a, u_a], w_a\;}_{x_a}$.

- The cycle inequalities (2) are defined for oriented cycles and involve the slack variables of their forward arcs with coefficient $+1$, and the slack variables of their backward arcs with coefficient -1.
- As a consequence, along a forward arc, the rounding benefit in the lower bound in (2) can force the slack y_a to increase and hereby immediately contribute to increasing the objective value. This is the right direction when the dual bound is supposed to increase.
- In contrast, along a backward arc, increasing its slack is relative to the upper bound u_a: If the value $u_a - (\ell_a + y_a)$ increases, then y_a decreases – and so does the objective function, which is *not* the actual intention.
- Hence, if a lower bound cycle inequality (2) enforces slack to be added, this risks to stay without any effect on the dual bound, whenever slack values along forward and backward arcs cancel out due to their opposite signs.

Thus, in order to identify strong lower bounds for the objective function – hence impose lower bounds on the slack values of groups of "expensive" arcs – we propose to also strive in particular for such cycle inequalities that prevent large slack values to just cancel out, hereby not contributing to the objective values. This is the case if the slack values of *all* arcs of the cycle γ appear in the valid inequality with the same sign, i.e., if the arcs of the cycle γ are forward arcs, i.e., if γ is a directed circuit.

We illustrate the possible benefit of considering the cycle inequalities for directed circuits in the example in Figure 1. Here, we set the period time $T = 10$. The graph G is planar and we consider the three oriented circuits γ_1, γ_2, and γ_3 that constitute the three finite faces of G (thus, a 2-basis or planar basis [11]), where all of them are oriented counter-clockwise. Moreover, we are going to deal with linear combinations of them, e.g., $\gamma_1 + \gamma_2 + \gamma_3$ is the outer oriented circuit, which is marked in green in Figure 1. This directed circuit represents the circulation of the trains of one line: One direction consists of the three bottom arcs, the opposite direction of the three top arcs. The two arcs in the terminus stations having the feasible interval $[1, 10]$ model the turnaround activities of the trains.

The two remaining arcs in the center region could, for instance, model single-track requirements [13]. Notice that such a configuration of single tracks close to both endpoints of a line could, e.g., be found at Line S2 of the Berlin S-Bahn network.

In order to ensure efficient operations, a major goal for a timetable in this example network is to operate the line with as few train units as possible. Equivalently, we minimize each slack time of any arc that models an activity of a vehicle. This applies to the arcs of $\gamma_1 + \gamma_2 + \gamma_3$ and we put weight $w_a = 10$ to these vehicle arcs. Moreover, slack time on any of the six non-turnaround vehicle arcs also involves extra waiting time for the passengers on board the train. Hence, for these six arcs, we even slightly increase their weights to $w_a = 11$. The two single-track arcs do not show any penalty weight, thus $w_a = 0$.

Now, let us consider the mixed-integer program (1) as well as its LP relaxation. On the one hand, in the integer optimum solution in particular the tension values $x = y + \ell$ along the outer cycle $\gamma_1 + \gamma_2 + \gamma_3$ have to sum up to an *integer* multiple of the period time. Indeed, one optimal solution reads

◼ **Table 1** Overview of the bounds of the cycle inequalities of the PESP instance in Figure 1.

Oriented circuits	lower bound	upper bound	integer values	forward	minimum arc weight
$\gamma_1,\ \gamma_3$	$\left\lceil \frac{1+1+1-7}{10} \right\rceil = 0$	$1 = \left\lfloor \frac{2+10+2-1}{10} \right\rfloor$	2	no	0
γ_2	$\left\lceil \frac{3+1+3+1}{10} \right\rceil = 1$	$2 = \left\lfloor \frac{6+7+6+7}{10} \right\rfloor$	2	yes	0
$\gamma_1 + \gamma_2,\ \gamma_2 + \gamma_3$	$\left\lceil \frac{1+1+3+1+3+1}{10} \right\rceil = 1$	$3 = \left\lfloor \frac{10+2+6+7+6+2}{10} \right\rfloor$	3	yes	0
$\gamma_1 + \gamma_2 + \gamma_3$	$\left\lceil \frac{12}{10} \right\rceil = 2$	$4 = \left\lfloor \frac{40}{10} \right\rfloor$	3	yes	10

$$x_a = \begin{cases} \ell_a + 8 = 9 & \text{for the westernmost arc } a, \\ \ell_a & \text{otherwise.} \end{cases}$$

On the other hand, for the LP relaxation it is well-known that its trivial optimal solution is just $x = \ell$ and $z = \frac{1}{T} \cdot (\Gamma x)$. Hence, we want to add valid inequalities in order to cut off this trivial fractional solution – and get as close as possible to the integer optimal solution.

One immediate way of doing so is to carefully select the cycle basis B in (1), and then to add the cycle inequalities (2) for the basic cycles. Traditionally, the choice of the cycle basis has been mainly motivated by the goal to keep as few as possible values for the integer variables z [14].

On this particular instance, it turns out that the three oriented cycles γ_1, γ_2, and γ_3 are the only ones that constrain their corresponding integer variables to only two values, see Table 1. Hence, these are the most attractive cycles in the well-established approach of allowing only few integer values for the integer variables. But when adding their cycle inequalities to the LP relaxation, its trivial optimum solution value persists. The same holds for five further valid inequalities. Only when adding the lower bound cycle inequality (2) for the directed circuit $\gamma_1 + \gamma_2 + \gamma_3$, the trivial optimum solution is cut off – and the dual bound is even pushed immediately to the integer optimum value. However, well-established separation heuristics [2, 16] are not guaranteed to consider this specific cycle, but we want to profit from its impact on the dual bound.

Hence, we propose to consider in particular the following simple oriented cycles for the separation of cycle inequalities, and for finding an integral cycle basis for (1):

1. heavy cycles, whose smallest weight is maximum,

2. forward cycles, i.e., without any backward arcs.

Let us shortly discuss why these two properties seem to be promising. Imagine that a lower bound of a cycle inequality implies a certain amount of slack to be distributed among the arcs of the cycle. Then there might be only little effect on the dual bound for the objective value if the slack can be concentrated on arcs that have only very small weight – there, the unavoidable slack somehow escapes[2] the objective function. Similarly, in the presence of backward arcs, on these, slack is relative to the *upper* bounds u_a of these arcs. Rounding effects in the cycle inequalities (2) thus tend to push the slack away from its upper bound, hence increase $u_a - (\ell_a + y_a)$, decrease y_a, and finally do *not* immediately support the dual bound to improve.

---

[2] Also during manual planning, it is a common saying that the *art* of periodic timetabling is to locate any unavoidable slack on those activities which are least important.

We will discuss theoretical properties of cycle bases exclusively composed of forward cycles in Section 3.2, and we will give a recipe to construct an integral forward cycle basis on a common type of event-activity networks in Section 3.3.

3.2 Theory of Forward Cycle Bases

Let $G = (V, A)$ be a digraph. The elements $\gamma \in \mathbb{Z}^A$ that satisfy flow conservation at every vertex, i.e.,

$$\forall v \in V : \sum_{a \in \delta^+(v)} \gamma_a = \sum_{a \in \delta^-(v)} \gamma_a,$$

form an abelian group, the *cycle space* $\mathcal{C}$ of G. In this language, an *oriented cycle* is an element $\gamma \in \mathcal{C}$ with $\gamma_a \in \{-1, 0, 1\}$ for all $a \in A$; we write $a \in \gamma$ iff $\gamma_a \neq 0$. Arcs $a \in \gamma$ with $\gamma_a = 1$ and $\gamma_a = -1$ are called *forward* and *backward*, respectively. A *forward cycle* is an oriented cycle γ containing only forward arcs, i.e., $\gamma \in \mathcal{C} \cap \{0, 1\}^A$.

The rank of $\mathcal{C}$ is the *cyclomatic number* μ. A set B of μ oriented cycles is called

(1) a *cycle basis* of G if B is a basis of the $\mathbb{R}$-vector space $\mathcal{C} \otimes \mathbb{R}$,

(2) an *undirected cycle basis* of G if B is a basis of the $\mathbb{F}_2$-vector space $\mathcal{C} \otimes \mathbb{F}_2$,

(3) an *integral cycle basis* of G if B is a basis of the abelian group $\mathcal{C}$,

(4) a *weakly fundamental cycle basis* of G if the cycles in B can be ordered in such a way that for $i \in \{2, \dots, \mu\}$, there is an arc $a \in \gamma_i$ with $a \notin \gamma_1 \cup \cdots \cup \gamma_{i-1}$,

(5) a *strictly fundamental cycle basis* of G if B is the set of fundamental cycles of some spanning forest F of G.

In this hierarchy of cycle bases, the implications $(5) \Rightarrow (4) \Rightarrow (3) \Rightarrow (2) \Rightarrow (1)$ hold [11].

▶ **Definition 3.** *A* forward cycle basis *is a cycle basis consisting exclusively of forward cycles.*

We prefer the term *forward cycles* over *directed cycles*, as the notion of *directed cycle basis* is already taken for a cycle basis consisting of arbitrary oriented cycles [11]. The existence of forward cycle bases is related to strong connectedness:

▶ **Theorem 4** ([26, 5]). *A digraph G has a forward cycle basis if and only if each 2-edge-connected component is strongly connected.*

In contrast to the general situation, even a strongly connected digraph does not necessarily admit a forward strictly fundamental cycle basis, see Example 10 in the appendix.

We now turn to the *minimum weight cycle basis problem*: Given arc weights $c \in \mathbb{R}_{\geq 0}^A$, find a cycle basis B such that its weight $c(B) := \sum_{\gamma \in B} \sum_{a \in \gamma} c_a$ is minimum. As mentioned in Section 3.1, finding minimum weight cycle bases has been proven useful for accelerating the branch-and-cut process in MIP solvers for (1). As $\mathcal{C} \otimes \mathbb{R}$ and $\mathcal{C} \otimes \mathbb{F}_2$ are vector spaces, the minimum weight cycle basis and minimum weight undirected cycle bases problems can be solved by the greedy algorithm on vector matroids. A polynomial-time algorithm can be constructed by restricting to a polynomially bounded set of cycles that are guaranteed to contain a minimum weight cycle basis, this is Horton's algorithm [10]. Recall that there are more efficient algorithms known [1]. For integral cycle bases, the complexity is unclear, whereas APX-hardness is known for the minimum weight weakly [24] resp. strictly [4] fundamental cycle basis problem .

▶ **Definition 5.** *Given $c \in \mathbb{R}_{\geq 0}^A$, the* minimum forward (undirected/integral/...) cycle basis problem *is to find a* forward *(undirected/integral/...) cycle basis B of minimum weight.*

Forward cycles in G together with sets of linearly independent forward cycles in $\mathcal{C} \otimes \mathbb{R}$ and $\mathcal{C} \otimes \mathbb{F}_2$ form a vector matroid, so that the greedy algorithm applies. Horton's algorithm can be adapted in such a way that it computes a minimum forward cycle basis or minimum forward undirected cycle basis in polynomial time [5].

However, we require *integral* cycle bases for the purpose of periodic timetabling. In practice, we observed that minimum (forward) undirected cycle bases are almost always integral, so that applying Horton's algorithm already solves the minimum weight integral cycle basis problem. We will also use this approach in Section 4.

3.3 ILTY cycles

We indicate the existence of forward integral cycle bases for event-activity networks with a special structure typical for periodic timetabling instances; we will call these networks *line-based*. A *line network* (N, L) is an undirected graph $N = (S, E)$ together with a set L of pairwise edge-disjoint simple paths whose union is E. We call S the set of *stations*, and L the set of *lines*.

▶ **Construction 6.** Given a line network (N, L), we construct a digraph G as follows:

(1) For each line $l \in L$ with sequence of stations $(s_1, \ldots, s_k)$, add

    (a) *driving arcs* $((s_i, \overrightarrow{l}, \text{dep}), (s_{i+1}, \overrightarrow{l}, \text{arr}))$ and $((s_{i+1}, \overleftarrow{l}, \text{dep}), (s_i, \overleftarrow{l}, \text{arr}))$ for $i \in \{1, \ldots, k-1\}$,

    (b) *dwell arcs* $((s_i, \overrightarrow{l}, \text{arr}), (s_i, \overrightarrow{l}, \text{dep}))$ and $((s_i, \overleftarrow{l}, \text{arr}), (s_i, \overleftarrow{l}, \text{dep}))$ for $i \in \{2, \ldots, k-1\}$,

    (c) *turnaround arcs* $((s_k, \overrightarrow{l}, \text{arr}), (s_k, \overleftarrow{l}, \text{dep}))$ and $(s_1, \overleftarrow{l}, \text{arr})), ((s_1, \overrightarrow{l}, \text{dep})$.

(2) For each station $s \in S$ and each pair (l, l') of distinct lines containing s, add *transfer arcs* $((s, \overrightarrow{l}, \text{arr}), (s, \overrightarrow{l'}, \text{dep}))$, $((s, \overrightarrow{l}, \text{arr}), (s, \overleftarrow{l'}, \text{dep}))$, $((s, \overleftarrow{l}, \text{arr}), (s, \overrightarrow{l'}, \text{dep}))$, and $((s, \overleftarrow{l}, \text{arr}), (s, \overleftarrow{l'}, \text{dep}))$.
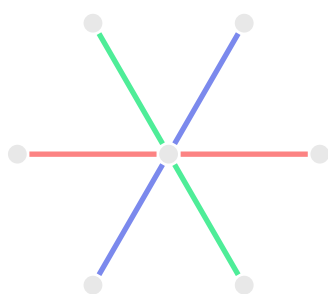
(a) Line network (N, L).
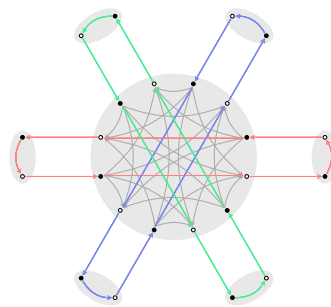
(b) Line-based event-activity network G.

■ **Figure 2** Example of Construction 6.

Construction 6 implicitly defines the vertices of G, each vertex is a triple consisting of a station in S, a line in L together with one of the direction markers $\leftarrow$ or $\rightarrow$, and the departure/arrival flag dep or arr. In particular, we can speak of the station or the line of a vertex, and conversely of vertices of a station or line. An example for Construction 6 on a star-shaped line network is depicted in Figure 2.
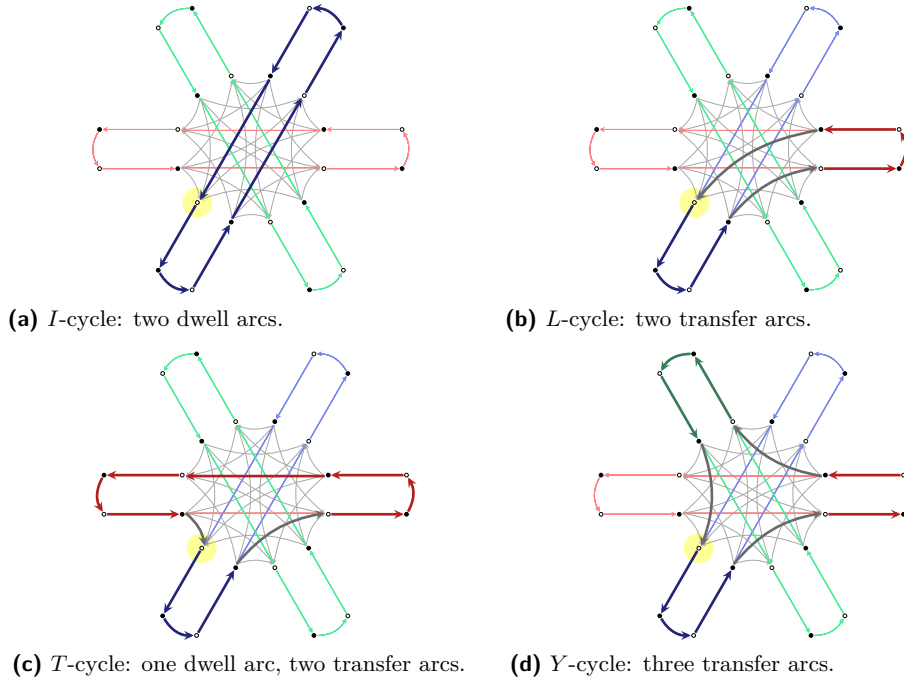
▶ **Definition 7.** *A digraph G is called* line-based *if it arises from a line network (N, L) via Construction 6.*

We will now define particularly simple types of a cycles in line-based digraphs.

▶ **Construction 8.** Let G be line-based network arising from $(N = (S, E), L)$, let $s \in S$ be a station. Construct the following forward cycles in G:

(1) *I-cycles* at s: forward cycles that contains no transfer arcs and at least one vertex of s,

(2) *L-cycles* at s: exactly two transfer arcs, both at s, and no dwell arcs at s,

(3) *T-cycles* at s: exactly two transfer arcs, both at s, and exactly one dwell arc at s,

(4) *Y-cycles* at s: exactly three transfer arcs, all of them at s, and no dwell arcs at s.

Intuitively, an *I*-cycle contains all driving, dwell and turnaround arcs associated to a single line. *L*- and *T*-cycles connect two lines, and *Y*-cycles connect three lines, all of them make use of transfers only at a single station. This family of cycles is depicted in Figure 3.

(a) *I*-cycle: two dwell arcs.

(b) *L*-cycle: two transfer arcs.

(c) *T*-cycle: one dwell arc, two transfer arcs.

(d) *Y*-cycle: three transfer arcs.

■ **Figure 3** *ILTY* cycles for the event-activity network in Figure 2 passing through a fixed vertex (marked in yellow).

The following is our main theorem about *ILTY* cycles, due to length restrictions, we refer to [18] for a proof.

▶ **Theorem 9.** *Let $(N = (S, E), L)$ be a line network, defining G via Construction 6. At each station $s \in S$, fix a vertex v of G at s, and let B_s denote the set of ILTY cycles at s passing through v. Then there is a set B' of forward cycles in G such that:*

(1) *B' projects to a strictly fundamental cycle basis of N,*

(2) *$\bigcup_{s \in S} B_s$ is a weakly fundamental basis of the subspace of the cycle space of G generated by all ITLY cycles,*

(3) *$B' \cup \bigcup_{s \in S} B_s$ is an integral cycle basis of G.*

4   Turnarounds in `R1L1`

The railway instances `RxLy` of the PESPlib are similarly structured [7, 17]. We will analyze the smallest instance `R1L1`, and to some extent reverse-engineer an underlying line network. The outcome is a modified instance `R1L1v`, where certain turnaround arcs have been added.

After removing four arcs with lower and upper bound 0, the network of `R1L1` becomes bipartite, vertices can hence be partitioned into arrival and departure events. This vertex labeling can be done in such a way that free arcs, i.e., arcs a with $u_a - \ell_a \geq T - 1 = 59$, always originate at an arrival and end at a departure. We interpret these arcs as transfer arcs. The remaining arcs a going from arrivals to departures have all $[\ell_a, u_a] = [1, 5]$, we take them as dwell arcs. We view the arcs from departures to arrivals as driving arcs. In fact, the network can be seen as a subnetwork of a line-based network as defined in Section 3.3.

If we now remove all transfer arcs, then the network decomposes into 110 directed simple paths, alternatingly consisting of driving and dwell arcs. We observe that for each such path, we find a path using exactly the same sequence of lower and upper bounds for the driving arcs in reverse order. Actually, this complies with the ordering of the activities as given in the text file describing the instance. This leads to a perfect matching of the directed paths, which we use to create 55 bidirectional lines. In the spirit of Construction 6, we do this by adding, in total, 110 turnaround arcs at both ends for each line. We call the resulting network `R1L1v`, where `v` stands for "vehicle", as vehicle turnarounds at the line ends have been modeled. `R1L1v` satisfies the hypothesis of Theorem 4, so that it admits a forward cycle basis. We will computationally evaluate several combinations of bounds and weights for the turnaround arcs and compare four cycle bases for `R1L1v` in Section 5.

5 Computational Results

For the network `R1L1v` described in Section 4, we compare four minimum turnaround times, seven weights (see Section 5.1.1), and four cycle bases, three of which are forward (see Section 5.1.2). Combining these, we obtain $4 \cdot 7 \cdot 4 = 112$ scenarios in total. We attack these scenarios with five primal strategies and one dual strategy (see Section 5.1.3) within the `ConcurrentPESP` solver [3] that computed the currently best primal and dual bounds for all PESPlib instances [6]. `ConcurrentPESP` invokes Gurobi 9.1 [9] as MIP solver. The computations were performed on an Intel Xeon E3-1270 CPU running at 3.80 GHz with 32 GB RAM, using up to 8 threads, with a wall time limit of 1 hour for each scenario.

5.1 Detailed Set-up

5.1.1 Bounds and Weights

Table 2 Parameters for turnaround arcs. The maximum weight in the original `R1L1` instance is $72\,523$, the arithmetic mean weight is $7\,388$.

Parameter	Values
minimum turnaround times ℓ_a	0, 5, 10, 15
turnaround weights w_a	0, 2500, 5000, 10000, 20000, 40000, 80000

The arc set of `R1L1v` consists of the arc set of `R1L1` plus 110 new turnaround arcs. For the turnaround arcs, we choose the same lower bound and the same weight from the values in Table 2, creating $4 \cdot 7 = 28$ PESP instances. All turnaround arcs are introduced as free arcs, so that we set $u_a := \ell_a + 59$. For the other arcs, we keep the bounds and weights as in `R1L1`. However, we also add the turnaround weight to all vehicle-related arcs, i.e., all driving and dwell arcs, in order to reflect the full vehicle rotation in the objective function.

5.1.2 Cycle Bases

We will consider the following four cycle bases on `R1L1v`:

(B1) `span`: a minimum integral cycle basis w.r.t. the cost $c_a := u_a - \ell_a$ for $a \in A$,

(B2) `forward span`: a minimum forward integral cycle basis w.r.t. $c_a := u_a - \ell_a$ for $a \in A$,

(B3) `forward bottleneck`: a forward integral cycle basis B maximizing $\sum_{\gamma \in B} \min_{a \in \gamma} w_a$,

(B4) `ILTY`: a forward integral cycle basis consisting of as many *ILTY* cycles as possible.

The minimum `span` basis (B1) is the classical approach to minimize the number of values the integer variables in (1) can attain. We motivated (B2) and (B3) in Section 3.1, (B4) comes from the construction in Section 3.3. Cycle bases (B1)-(B3) can be computed by (a modification of) Horton's algorithm [10], the result always turns out to be not only a minimum undirected cycle basis, but also integral. Note that unlike the other bases, (B3) needs to be recomputed for every choice of weight for the turnaround arcs, and we need to consider bottleneck shortest paths for Horton's algorithm. For (B4), as `R1L1v` is only a subnetwork of a line-based network, we follow an analytic approach instead of the synthetic one pursued in Construction 8 to construct *ILTY* cycles: We seek for single lines, pairs and triples of lines, and construct all possible *ILTY*-shaped cycles whenever they exist in the network. This produces a subspace of the cycle space of dimension 2823, however, its codimension is only 9. We therefore complete the *ILTY* cycles with 9 cycles from the `forward span` basis.

5.1.3 Solution Strategies

Table 3 Solution strategies for the computational experiments.

Strategy	MIP	Initial solution	Ignore light arcs	Other heuristics
`complete`	✓		✓	✓
`mip`	✓			
`mip-start`	✓	✓		
`mip-ignore`	✓		✓	
`mip-ignore-start`	✓	✓	✓	
`dual`	✓	✓		

As `ConcurrentPESP` contains a variety of PESP algorithms, we singled out 6 solution methods, summarized in Table 3. As we want to compare cycle bases, we concentrate on the MIP component of the solver. For `mip-start`, `mip-ignore-start`, and `dual`, the current PESPlib incumbent for `R1L1` with weighted slack 29 894 745 [6, 17] is given the solver as an initial solution. Ignoring light arcs means to solve PESP on a subnetwork (with the same strategy) as described in [7, 3]. Other heuristics include, e.g., the modulo network simplex method [22, 8]. The emphasis for `mip-start` lies on the primal bound, whereas `dual` focuses on improving the dual bound, and additionally invokes flip cutting planes [16].

5.2 Results

Figure 4 displays the results of our computations in the Pareto sense for the weighted slack and the number of vehicles as objectives. The weighted slack is computed on the original instance `R1L1`, i.e., omitting turnaround arcs, and hence is the passenger-oriented component. The number of vehicles is obtained as the sum of the driving, dwell and turnaround times (periodic tensions) for each line divided by the period time $T = 60$. Clearly, higher turnaround times require more vehicles.
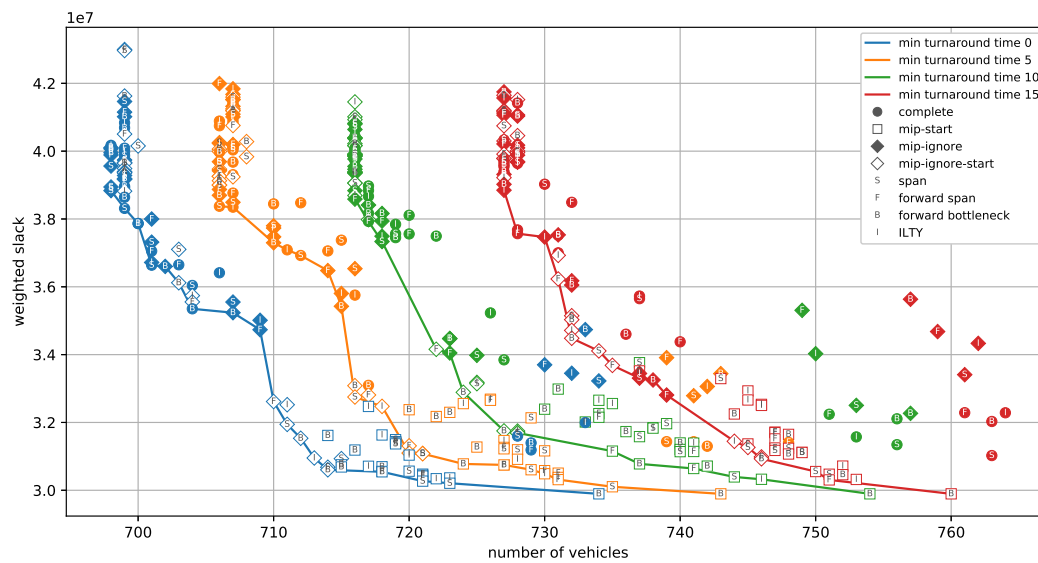
Figure 4 Overview of all results for the strategies `complete`, `mip-start`, `mip-ignore`, and `mip-ignore-start`. The solid lines depict the Pareto front for the four minimum turnaround times. The marker shape indicates the strategy, the inscribed letter stands for the cycle basis. A higher resolution version of the chart is available at `https://www.zib.de/lindner/R1L1v-hi-res.pdf`.

For the `mip` strategy, which runs Gurobi without an initial solution and no further help from other heuristics, *all* solutions have a weighted slack of more than 48 000 000, so that the solution quality is much worse than for the other strategies. For the other strategies, the picture is quite diffuse: Each of the other four primal strategies and each of the cycle bases produces at least once a non-dominated solution, and this holds even for each minimum turnaround time individually. However, there is a tendency that starting with the PESPlib incumbent maintains the small weighted slack, while starting without initial solution brings the number of vehicles down.

To assess the impact of the cycle bases, we compare these for each strategy with a hypothetical cycle basis that attains the best weighted slack (or number of vehicles) among all four cycle bases for each instance with that strategy. Table 4 reports the relative gap in percent, averaged over all 28 combinations of lower bounds and weights.

Table 4 Average relative gaps in percent. The first column per basis is for the weighted slack, the second column is for the number of vehicles. E.g., 2.00 for `complete` in the leftmost `span` column means that the `span` basis produce a weighted slack that is in average 2% worse than a hypothetical cycle basis that takes the best slack among all four cycle bases with the `complete` strategy.

strategy	span		fwd bottleneck		forward span		ILTY	
`complete`	2.00	0.13	2.16	0.12	2.59	0.14	2.32	0.17
`mip`	0.12	0.07	5.47	0.44	3.42	0.24	4.19	1.05
`mip-start`	1.16	0.57	1.73	0.12	0.90	0.43	1.89	0.48
`mip-ignore`	1.26	0.12	1.49	0.12	2.15	0.05	1.49	0.09
`mip-ignore-start`	1.32	0.08	1.05	0.06	1.53	0.06	1.27	0.05
`dual`	17.66		4.81		3.35		1.44	

It turns out that the traditional minimum span basis, which is allowed to contain backward arcs, performs best in terms of weighted slack for the strategies that were not given a high quality timetable as input. For the other strategies, a forward basis was better. However, with the exception of the `mip` strategy, the quality differences are mostly only minor, and the same holds for the number of vehicles. We can conclude that on the primal side, forward cycle perform similar to the minimum span basis.

What is however striking is the dual side: After the time limit of one hour, the minimum span basis is far worse than all forward cycle bases, and the `ILTY` basis is the clear winner.

Note that the best lower bound on the number of vehicles is quickly obtained by summing up the lower bound of each cycle inequality (2) for the I-cycles of each line.

5.3   A New Dual Incumbent for `R1L1`

Motivated by the dual performance of the forward cycle basis, we try to compute a new dual bound for the original PESPlib instance `R1L1`. When the minimum turnaround time is 0 and the turnaround weights are 0, every timetable for `R1L1` is feasible for `R1L1v` and vice versa, and the weighted slacks do not change. In this setting, we can hence use the forward cycle bases on `R1L1v` to compute dual bounds for `R1L1`. One can check that forward cycle bases do not exist on `R1L1`, as it fails to satisfy the hypothesis of Theorem 4.

Table 5 A new dual incumbent for `R1L1` through `R1L1v`.

instance	cycle basis	dual bound
`R1L1v`	span	20 638 013
`R1L1v`	forward span	20 609 801
`R1L1v`	forward bottleneck	20 591 564
`R1L1v`	ILTY	20 901 883
`R1L1`	span	20 693 118

Table 5 compares the dual bounds after 24 hours wall time on up to 6 threads obtained by the four cycle bases on `R1L1v`, and additionally by the `span` basis on `R1L1`. As bounds basically stop moving after one hour in Gurobi, we switched to CPLEX 12.10 for this particular computational experiment, which performs better in the long run. We again use flip inequalities as source for cutting planes. Although the `span` basis becomes better, `ILTY` on `R1L1v` produces the best dual bound, although `R1L1v` is a larger instance and the cycle space dimension has increased by 110.

6   Conclusion and Outlook

Minimum forward cycle bases are competitive when seeking high quality solutions to PESP instances, and superior when computing of dual bounds. The $ILTY$ cycles, that can be constructed on line-based event-activity networks, are particularly strong: They allow for better dual bounds on the PESPlib instance `R1L1`, although the computation is carried out on the larger `R1L1v`. The natural question is whether this strength extends to other PESP instances. So far the PESPlib does not contain instances with specified turnarounds. We submitted a realization of `R1L1v` with minimum turnaround time 10 and turnaround weight 5000, and this instance is now part of the PESPlib. A similar procedure is possible for all `RxLy` PESPlib instances.

─── **References** ───

1   E. Amaldi, C. Iuliano, T. Jurkiewicz, K. Mehlhorn, and R. Rizzi. Breaking the $o(m^2n)$ barrier for minimum cycle bases. In A. Fiat and P. Sanders, editors, *Algorithms - ESA 2009, 17th Annual European Symposium, Copenhagen, Denmark, September 7-9, 2009. Proceedings*, volume 5757 of *Lecture Notes in Computer Science*, pages 301–312. Springer, 2009. `doi:10.1007/978-3-642-04128-0_28`.

2   R. Borndörfer, H. Hoppmann, M. Karbstein, and N. Lindner. Separation of cycle inequalities in periodic timetabling. *Discrete Optimization*, 35:100552, 2020. `doi:10.1016/j.disopt.2019.100552`.

3   R. Borndörfer, N. Lindner, and S. Roth. A concurrent approach to the Periodic Event Scheduling Problem. *Journal of Rail Transport Planning & Management*, 15:100175, 2020. Best Papers of RailNorrköping 2019. `doi:10.1016/j.jrtpm.2019.100175`.

4   G. Galbiati, R. Rizzi, and E. Amaldi. On the approximability of the minimum strictly fundamental cycle basis problem. *Discrete Applied Mathematics*, 159(4):187–200, 2011. `doi:10.1016/j.dam.2010.10.014`.

5   P. M. Gleiss, J. Leydold, and P. F. Stadler. Circuit bases of strongly connected digraphs. *Discussiones Mathematicae Graph Theory*, 23(2):241, 2003. `doi:10.7151/dmgt.1200`.

6   M. Goerigk. PESPlib – A benchmark library for periodic event scheduling, 2012. URL: `http://num.math.uni-goettingen.de/~m.goerigk/pesplib/`.

7   M. Goerigk and C. Liebchen. An improved algorithm for the periodic timetabling problem. In *ATMOS*, volume 59 of *OASICS*, pages 12:1–12:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

8   M. Goerigk and A. Schöbel. Improving the modulo simplex algorithm for large-scale periodic timetabling. *Computers & Operations Research*, 40(5):1363–1370, 2013.

9   Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2021. URL: `https://www.gurobi.com`.

10 J. D. Horton. A polynomial-time algorithm to find the shortest cycle basis of a graph. *SIAM J. Comput.*, 16(2):358–366, 1987. `doi:10.1137/0216026`.

11 T. Kavitha, C. Liebchen, K. Mehlhorn, D. Michail, R. Rizzi, T. Ueckerdt, and K. A. Zweig. Cycle bases in graphs characterization, algorithms, complexity, and applications. *Computer Science Review*, 3(4):199–243, 2009. `doi:10.1016/j.cosrev.2009.08.001`.

12 C. Liebchen. *Periodic timetable optimization in public transport*. PhD thesis, Technische Universität Berlin, 2006.

13 C. Liebchen and R. H. Möhring. The Modeling Power of the Periodic Event Scheduling Problem: Railway Timetables — and Beyond. In F. Geraets, L. Kroon, A. Schöbel, D. Wagner, and C. D. Zaroliagis, editors, *Algorithmic Methods for Railway Optimization*, Lecture Notes in Computer Science, pages 3–40, Berlin, Heidelberg, 2007. Springer. `doi:10.1007/978-3-540-74247-0_1`.

14 C. Liebchen and L. Peeters. Integral cycle bases for cyclic timetabling. *Discrete Optimization*, 6(1):98–109, 2009. `doi:10.1016/j.disopt.2008.09.003`.

15 C. Liebchen and E. Swarat. The Second Chvatal Closure Can Yield Better Railway Timetables. In M. Fischetti and P. Widmayer, editors, *8th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems (ATMOS'08)*, volume 9 of *OpenAccess Series in Informatics (OASIcs)*, Dagstuhl, Germany, 2008. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. ISSN: 2190-6807. `doi:10.4230/OASIcs.ATMOS.2008.1580`.

16 N. Lindner and C. Liebchen. Determining All Integer Vertices of the PESP Polytope by Flipping Arcs. In D. Huisman and C. D. Zaroliagis, editors, *20th Symposium on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2020)*, volume 85 of *OpenAccess Series in Informatics (OASIcs)*, pages 5:1–5:18, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. ISSN: 2190-6807. `doi:10.4230/OASIcs.ATMOS.2020.5`.

17   N. Lindner and C. Liebchen. Timetable Merging for the Periodic Event Scheduling Problem. Technical Report 21-06, Zuse Institute Berlin, 2021. URL: `https://nbn-resolving.org/urn:nbn:de:0297-zib-81587`.

18   N. Lindner, C. Liebchen, and B. Masing. Constructing forward integral cycle bases for periodic timetabling, in preparation.

19   T. Lindner. *Train Scheduling in Public Rail Transport*. PhD thesis, Technische Universität Braunschweig, 2000.

20   K. Nachtigall. Cutting Planes for a Polyhedron Associated with a Periodic Network. Technical Report 112-96/17, Deutsches Zentrum für Luft- und Raumfahrt, 1996.

21   K. Nachtigall. *Periodic Network Optimization and Fixed Interval Timetables*. Habilitation thesis, Universität Hildesheim, 1998.

22   K. Nachtigall and J. Opitz. Solving Periodic Timetable Optimisation Problems by Modulo Simplex Calculations. In Matteo Fischetti and Peter Widmayer, editors, *8th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems (ATMOS'08)*, volume 9 of *OpenAccess Series in Informatics (OASIcs)*, Dagstuhl, Germany, 2008. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.

23   M. A. Odijk. Construction of periodic timetables, part 1: A cutting plane algorithm. Technical Report 94-61, TU Delft, 1994.

24   R. Rizzi. Minimum Weakly Fundamental Cycle Bases Are Hard To Find. *Algorithmica*, 53(3):402–424, 2009. `doi:10.1007/s00453-007-9112-8`.

25   P. Serafini and W. Ukovich. A mathematical model for periodic scheduling problems. *SIAM Journal on Discrete Mathematics*, 2(4):550–581, 1989.

26   P. Seymour and C. Thomassen. Characterization of even directed graphs. *Journal of Combinatorial Theory, Series B*, 42(1):36–45, 1987. `doi:10.1016/0095-8956(87)90061-X`.

A   Non-Existence of Forward Strictly Fundamental Cycle Bases

(a) Graph.   **(b)** γ_1.   **(c)** γ_2.   **(d)** γ_3.   **(e)** γ_4.   **(f)** γ_5
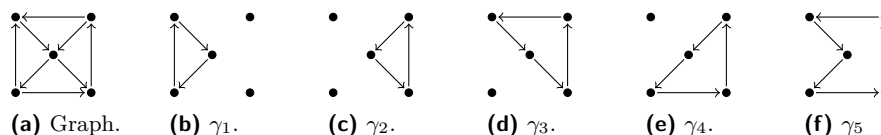
Figure 5 Strongly connected graph without a forward strictly fundamental cycle basis.

▶ **Example 10.** Consider the strongly connected digraph G in Figure 5a. There are five forward cycles $\gamma_1, \ldots, \gamma_5$ (cf. Figure 5), note that $\gamma_2 + \gamma_5 = \gamma_3 + \gamma_4$. The cyclomatic number of G is 4, and the combinations of three of the cycles $\gamma_2, \ldots, \gamma_5$ with γ_1 form all forward cycle bases. A spanning tree F of G has four co-tree arcs, in a strictly fundamental cycle basis for F, each co-tree arc must appear in exactly one cycle. However, in any forward cycle basis, we find at most three exclusive arcs, so that no forward cycle basis is strictly fundamental.