Solving the Electric Bus Scheduling Problem by an Integrated Flow and Set Partitioning Approach

Ralf Borndörfer
Zuse Institute Berlin, Germany

Andreas Löbel
IVU Traffic Technologies AG, Berlin, Germany

Fabian Löbel[1]
Zuse Institute Berlin, Germany

Steffen Weider
IVU Traffic Technologies AG, Berlin, Germany

Abstract

Attractive and cost-efficient public transport requires solving computationally difficult optimization problems from network design to crew rostering. While great progress has been made in many areas, new requirements to handle increasingly complex constraints are constantly coming up. One such challenge is a new type of resource constraints that are used to deal with the state-of-charge of battery-electric vehicles, which have limited driving ranges and need to be recharged in-service.

Resource constrained vehicle scheduling problems can classically be modelled in terms of either a resource constrained (multi-commodity) flow problem or in terms of a path-based set partition problem. We demonstrate how a novel integrated version of both formulations can be leveraged to solve resource constrained vehicle scheduling with replenishment in general and the electric bus scheduling problem in particular by Lagrangian relaxation and the proximal bundle method.

2012 ACM Subject Classification Mathematics of computing → Mathematical optimization; Mathematics of computing → Network flows; Mathematics of computing → Integer programming; Mathematics of computing → Linear programming; Applied computing → Transportation

Keywords and phrases Electric Bus Scheduling, Electric Vehicle Scheduling, Non-linear Charging, Multi-commodity Flow, Set Partition, Lagrangian Relaxation, Proximal Bundle Method

Digital Object Identifier 10.4230/OASIcs.ATMOS.2024.11

Funding This work has been conducted within the Research Campus MODAL funded by the German Federal Ministry of Education and Research (BMBF) (fund number 05M20ZBM).

1 Introduction

Public transport operators are ramping up the electrification of their bus fleets. Operators in major German cities like Berlin, Hamburg or Munich have pledged to fully electrify their public transport systems by 2030 [25]. Moreover, starting in 2026, it is a legal requirement that 65% of new acquisitions have to have a clean drive train [30]. The European electric bus market is dominated by battery-powered vehicles, depot chargers and fast opportunity chargers at selected terminals [31, 5, 9, 32].

Deploying electric buses has to be planned around their complex energy-cycle. Unlike their diesel counterparts, which can usually drive for an entire day and be fully refueled within minutes upon returning to the depot, electric buses have limited driving ranges and significant recharging times. These limitations are usually exacerbated during summer and winter [31], for instance, enabling air conditioning may reduce a driving range of 250 km down

[1] Corresponding author.

to 175 km [35]. As such, electric buses either require short schedules or pre-planned detours and downtime throughout the day to recharge. Moreover, the physics behind recharging batteries lead to non-linear energy models [27], which must be considered for electric vehicle scheduling problems to ensure solutions are actually energy-feasible [24, 26, 22].

This paper is structured as follows: In Section 2 we define the electric bus scheduling problem and in Section 3 we briefly review solution approaches (with non-linear charging) from the literature. In Section 4 we provide a generalized formulation for resource constrained vehicle scheduling by integrating the common flow and path-based models presented in Section 2. In Section 5 we outline how to solve this new formulation leveraging Lagrangian relaxation and the so-called proximal bundle method. For the sake of brevity, we will often refer to older publications for details. Since this algorithm has been in commercial use at a number of public transport operators with partially electrified fleets for a few years now, we can not provide an open source implementation. Finally, we conclude with some computational results in Section 6.

2 Problem Description

Attractive and efficient public transport is contingent on high quality solutions to a number of strategic and operational planning steps, from infrastructure planning, line planning, and timetabling to vehicle scheduling, duty scheduling, crew rostering, and finally real-time disposition [34]. Due to the high computational complexity of each planning task, they are often solved sequentially, even though there are some feedback relationships.

For this paper, we consider the vehicle scheduling planning step for bus systems with (partially) battery-electric fleets, although we believe our results generalize to any electric vehicle scheduling or routing problem. We assume that the infrastructure, bus lines and the timetable have already been fixed, which yield a set of *timetabled passenger trips* $\mathcal{T}$. A *trip* $\tau \in \mathcal{T}$ is the activity of servicing a single repetition of a line from its first to its last stop or terminal. For example, if a bus line has a periodicity of five minutes, then it admits twelve trips per hour. The back direction of a line for this purpose is considered as a separate line.

Each trip needs to be serviced by a bus and any individual bus can not service any trips that happen simultaneously. Moreover, if a bus is scheduled to service two trips in order, it needs to be able to get from the end terminal of the first trip to the start terminal of the second in time to comply with the given fixed timetable. Generally, the set of trips $\mathcal{T}$ together with a relation $\prec$ giving feasible connections called *turns* or *deadheads* can be thought of as a partially ordered set.

The classic (non-electric) *bus scheduling problem* (*BSP*) in its simplest form is to find a cost optimal partition of the trips into chains (subsets of ordered trips), which we call a *vehicle schedule*. An individual chain of trips, i.e., a sequence of trips that can be serviced by the same bus, is a *vehicle course*. The objective function would generally minimize the total required fleet size, i.e., the number of vehicle courses, but also the overall operational expenses.

We are further given a set of depots $\mathcal{D}$ and vehicle types $\mathcal{V}$. Each course has to be assigned a depot the bus has to begin and end service at, and a vehicle type which determines operational costs, but also which trips, deadheads, depots or potentially other infrastructure are accessible. Articulated buses or double-deckers may not fit through every road but some trips have to be serviced by vehicles with a higher capacity for passengers. Moreover, in the electric setting (Section 2.1), we may have vehicle types with different driving ranges, but larger batteries incur higher deployment costs.

We can model *BSP* as a multi-commodity flow problem on a directed acyclic graph where pairs of vehicle type and depot $K = \mathcal{V} \times \mathcal{D}$ serve as the commodities, which we call *plan types*. Every trip admits a node and we have an arc of the form $(\tau_i, \tau_j) \in A$ if and only if $\tau_i \prec \tau_j$, that is, τ_j can be serviced after trip τ_i by the same bus. For every depot $d \in \mathcal{D}$ we add a source node d^{out} and a sink node d^{in} and connect them by pull-out and pull-in arcs to every trip. $K(a)$ then denotes the plan types admissible on arc a, which can be used to control access of particular types to trips. By restricting the pull-in and pull-out arcs at a depot to only the compatible plan types, we further enforce that buses actually return to the depot that they started their course at.

A vehicle course for type $v \in \mathcal{V}$ is then a path between the source and sink nodes of a depot d whose arcs all permit the plan type (v, d). The cost of a vehicle course is simply the sum over arc costs c_a^k of its plan type, which reflect operational expenses. A (binary) multi-commodity flow of minimum cost on this graph then yields a minimum cost vehicle schedule.

Moreover, we have so-called *vehicle-mix constraints* which impose lower and upper bounds on how many courses may be assigned to particular plan types. This is because depots can usually only accommodate a certain number of buses of any particular type, or operators may insist that the fleet composition stay within some parameter. They are given by a family of plan type subsets $\bar{K} \subset 2^K$ and for each $\mathcal{K} \in \bar{K}$, we have bounds $\ell_{\mathcal{K}}$ and $u_{\mathcal{K}}$, as well as coefficients $\kappa_{\mathcal{K}}^k$ per $k \in \mathcal{K}$ such that the weighted sum over all courses of those plan types has to be within $[\ell_{\mathcal{K}}, u_{\mathcal{K}}]$.

In this *BSP* model, trips are generally connected to all reachable trips which happen later within the planning horizon, so a large fraction of the deadheads are *long* in the sense that operators prefer a bus assigned to such a deadhead makes a stopover at a parking facility or depot, where no driver has to be paid to watch over the idle vehicle. In the worst-case, the total number of deadheads is $|\mathcal{T}|\,(|\mathcal{T}|+1)/2$, therefore, long arcs are either dynamically generated on demand while solving *BSP* [20], or they are modeled implicitly via *timelines* (cf. [14, 12]): The planning horizon is discretized and for every time step and parking spot, a node is added to the graph. The nodes of a spot are connected by *idling arcs* in order and there are pull-out and pull-in deadheads between the trips and appropriate timeline nodes. Every long deadhead is then pruned as it now corresponds to a path between a pull-in and a pull-out along a timeline, which decreases the total number of arcs on instances of relevant size [12]. For an example *BSP* graph with five trips, two depots and one parking spot see Figure 1.

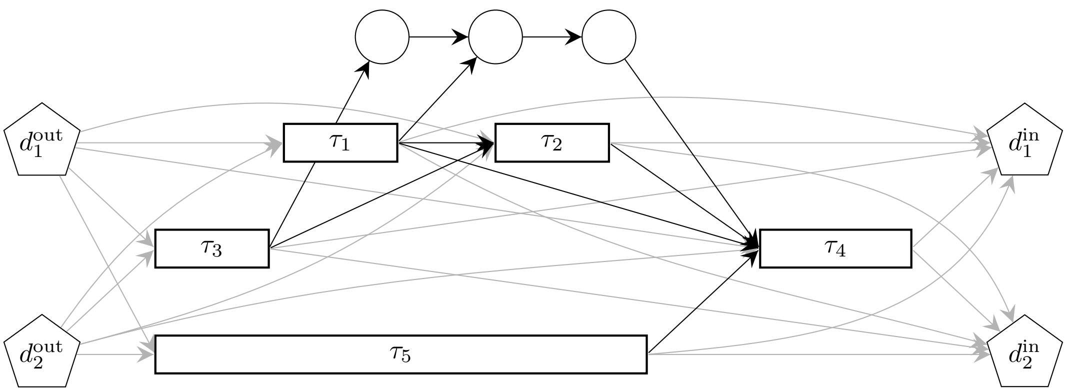

Figure 1 Example *BSP* Graph with five trips, two depots and one parking spot timeline.

The grey arcs are the depot pull-in and pull-out deadheads which only permit plan types of the associated depot. The trip nodes are elongated to indicate their relative start and end times. There is no deadhead from τ_3 to τ_1 as there is no way to make the connection in time. The long deadhead between τ_3 and τ_4 has been pruned in favor of the parking spot timeline on top of the graph, which of course needs larger graphs to be of any advantage. If some trips or deadheads may only be traversed by particular bus types, they must permit only the corresponding plan types, so that the respective nodes and arcs are no longer part of the network of those commodities.

Formulating this multi-commodity flow model of the non-electric *BSP* as an integer linear program, we obtain

$$(BSP) \quad \min \quad \sum_{a\in A}\sum_{k\in K(a)} c_a^k x_a^k \tag{1}$$

$$\text{s.t.} \quad \sum_{a\in\delta^{\text{in}}(n):\,K(a)\ni k} x_a^k - \sum_{a\in\delta^{\text{out}}(n):\,K(a)\ni k} x_a^k = 0 \qquad \forall n\in N\setminus\mathcal{D},\ k\in K \tag{2}$$

$$\sum_{a\in\delta^{\text{out}}(\tau)}\sum_{k\in K(a)} x_a^k = 1 \qquad \forall \tau\in\mathcal{T} \tag{3}$$

$$\sum_{a\in\delta^{\text{out}}(n)}\sum_{k\in K(a)} x_a^k \le 1 \qquad \forall n\in N\setminus(\mathcal{D}\cup\mathcal{T}) \tag{4}$$

$$\ell_{\mathcal{K}} \le \sum_{(v,d)\in\mathcal{K}} \kappa_{\mathcal{K}}^{(v,d)} \sum_{a\in\delta^{\text{out}}(d^{\text{out}})} x_a^{(v,d)} \le u_{\mathcal{K}} \qquad \forall \mathcal{K}\in\bar{K} \tag{5}$$

$$x_a^k \in \{0,1\} \quad \forall a\in A,\ k\in K(a) \tag{6}$$

where N denotes the entire set of nodes, i.e., it contains $\mathcal{T}$, the depot source and sink nodes and all timeline nodes. Furthermore, $\delta^{\text{in}}(n)$ denotes the set of incoming and $\delta^{\text{out}}(n)$ the set of outgoing deadheads at node n. The binary variables x_a^k indicate whether arc a is selected or *active* for plan type k, i.e., whether a bus of the corresponding type and housed at the corresponding depot traverses it. (2) are flow conservation constraints per commodity, which propagate the selected vehicle type and depot along the flow belonging to a vehicle course. (3) enforces that every trip is covered exactly once and (4) ensures that a parking spot can be used by at most one bus at the same time. (5) are the vehicle-mix constraints.

Note that if $|K| \ge 2$, *BSP* is NP-hard even without any vehicle-mix constraints [1]. Further note that *BSP* is a special case of the vehicle scheduling or routing problem, where vehicles have to visit a set of customers within pre-defined time windows to perform some task of a given duration. The trips correspond to customers with fixed and tight time windows.

Let $M^{\text{F}}x = b$ denote (2) - (5) of the flow formulation in an appropriate matrix notation.

2.1 The Bus Scheduling Problem with Electric Vehicles

The *electric bus scheduling problem* (*EBSP*) extends the *BSP* such that some or all of the bus types are powered by an electric battery. We collect the corresponding plan types in K^E and normalize all battery capacities and energy consumption to a relative driving range in $[0,1]$. Every deadhead arc admits an energy consumption e_a^k per electric plan type, including the consumption of its target trip. We track the remaining driving range via variables y_a at the beginning of every arc, just after the source trip.

Charging takes place at a limited number of charger slots $\mathcal{S}$, so we introduce timelines to track when they are occupied by a bus. We denote those *recharge nodes* s_i by $\bar{\mathcal{S}}$ and add them to N. The corresponding timeline arcs $a(s,i) = (s_{i-1}, s_i)$ are called *recharge arcs* and we denote the set of all recharge arcs by A^E. Along each we can replenish an amount of driving range given by a function $\Delta\zeta_s^k(y_{a(s,i)}, \theta)$ where $y_{a(s,i)}$ is the charge state at the beginning of the recharge arc $a(s,i)$ and θ is the step size of the time discretization. The *charge increment function* $\Delta\zeta_s^k$ depends on the technology employed at charger slot s and the vehicle type of the plan type k. Note that if we are given a charge curve ζ as most of the literature on *EBSP* assumes, that is, a function mapping time spent charging an initially empty battery to the resulting state-of-charge, then it relates to the increment function via $\Delta\zeta(y, \theta) = \zeta(\zeta^{-1}(y) + \theta) - y$. For a homogeneous time step size θ we can just write $\Delta\zeta(y)$.

This yields the, in general non-linear, mixed-integer program

$$(EBSP) \quad \min \sum_{a \in A,\, k \in K(a)} c_a^k x_a^k \tag{7}$$

$$\text{s.t.} \quad M^{\mathrm{F}} x = b \tag{8}$$

$$\sum_{k \in K^E(a)} x_a^k \geq y_a \qquad \forall a \in A \setminus A_{\mathcal{D}}^{\mathrm{out}} \tag{9}$$

$$\sum_{k \in K^E(a)} x_a^k = y_a \qquad \forall a \in A_{\mathcal{D}}^{\mathrm{out}} \tag{10}$$

$$\sum_{\substack{a \in \delta^{\mathrm{in}}(n) \\ k \in K^E(a)}} e_a^k x_a^k = \sum_{a \in \delta^{\mathrm{in}}(n)} y_a - \sum_{a \in \delta^{\mathrm{out}}(n)} y_a \qquad \substack{\forall n \in N \\ n \notin \mathcal{D} \cup \bar{\mathcal{S}}} \tag{11}$$

$$\sum_{\substack{a \in \delta^{\mathrm{in}}(s_i), \\ k \in K^E(a)}} e_a^k x_a^k - \sum_{k \in K^E(a(s,i))} \varphi_{a(s,i)}^k = \sum_{a \in \delta^{\mathrm{in}}(s_i)} y_a - \sum_{a \in \delta^{\mathrm{out}}(s_i)} y_a \qquad \forall a(s,i) \in A^E \tag{12}$$

$$\varphi_{a(s,i)}^k = \Delta\zeta_s^k(y_{a(s,i)}) x_{a(s,i)}^k \qquad \substack{\forall a(s,i) \in A^E, \\ k \in K^E(a(s,i))} \tag{13}$$

$$x_a^k \in \{0,1\} \qquad \forall a \in A,\ k \in K(a) \tag{14}$$

$$y_a \geq 0 \qquad \forall a \in A \tag{15}$$

where $A_{\mathcal{D}}^{\mathrm{out}}$ denotes the set of pull-out arcs at depot nodes, i.e., those arcs that can open new vehicle courses. We retain the *BSP* constraints in (8), but on the graph including charge slot timelines (an example graph would still look like the one in Figure 1, except the timeline may belong to a charge slot). (9) enforces that only active flow-carrying arcs can also have non-zero charge states while (10) requires buses to start service with a full battery. (11) and (12) propagate charge states along active arcs as an energy flow depending on whether any incoming arc is a recharge arc. (13) gives the amount of restored driving range on active recharge arcs depending on the incoming charge state. (14) and (15) are the variable domains. To make (*EBSP*) a linear program we have to linearize the constraint (13), where we refer to our contributions [21] and [22].

2.2 A Set Partition Formulation

It is well-known that *BSP* can be formulated as a set partition problem by applying Dantzig-Wolfe decomposition to the multi-commodity flow formulation. For $k = (v, d) \in K$, let P_k denote all $d^{\mathrm{out}}, d^{\mathrm{in}}$-paths admissible for vehicle type v on the *BSP* graph. Further, let $P = \cup_{k \in K} P_k$. Then, assuming a suitable cost vector $c \in \mathbb{R}^P$ (usually the sum over the arcs on the path), a formulation equivalent to (*BSP*) is

$$\min \sum_{p \in P} c_p x_p \tag{16}$$

$$\text{s.t.} \quad \sum_{p \in P: p \ni \tau} x_p = 1 \qquad \forall \tau \in \mathcal{T} \tag{17}$$

$$\sum_{p \in P:\ p \ni n} x_p \leq 1 \qquad \forall n \in N \setminus (\mathcal{D} \cup \mathcal{T}) \tag{18}$$

$$\ell^{\mathcal{K}} \leq \sum_{k \in \mathcal{K}} \sum_{p \in P_k} x_p \leq u^{\mathcal{K}} \qquad \forall \mathcal{K} \in \bar{K} \tag{19}$$

$$x \in \{0,1\}^P \tag{20}$$

In theory, it is straightforward to turn this into a formulation for *EBSP*: Let P be the set of all *energy-feasible* paths. A path on the *BSP* graph is energy-feasible if we can insert recharge events such that the battery is never fully depleted and all trips on the path can still be serviced as scheduled. Note that finding a cost-optimal recharge schedule for a given fixed sequence of trips is an instance of the NP-hard *fixed route vehicle charging problem* [24], while testing whether such a sequence is energy-feasible by just charging for as much as possible is polynomially solvable [4]. Further note that the set partition formulation is straightforward to generalize to any resource constrained vehicle scheduling problem with replenishment, like railway operation with maintenance scheduling.

Due to the large number of variables, which in the worst case is one per path on the vehicle scheduling graph, column generation lends itself as the go-to solving approach. The pricing problem is then a *resource constrained shortest path problem with replenishment*, i.e., we need to find resource-feasible paths on the vehicle scheduling graph with negative reduced costs. These paths have to be fit with a cost-optimal resource restoration schedule, which for *EBSP* generally involves evaluating non-linear $\Delta\zeta$.

3 Solution Approaches in the Literature

Electric vehicle routing and scheduling is an active area of research attracting an immense amount of attention. We therefore restrict this literature review to contributions presenting solving approaches for the electric vehicle scheduling problem that can handle non-linear charging explicitly. For an extensive survey on electric vehicle routing and scheduling we refer to [5] and on electric bus scheduling see [28].

An energy state expansion model is proposed in [33], where for each step of a charge state discretization, every node of the vehicle scheduling graph is duplicated. The deadhead arcs connect nodes of appropriate charge states with each other and if there is a recharge window, such a connection can go from a lower to a higher charge state, so $\Delta\zeta$ can be evaluated explicitly per recharge arc. The column generation pricing problem is then a classic shortest path problem on this energy-state-expanded graph. The column generation itself solves the Lagrangian relaxation of a path-based set cover formulation in combination with a rounding heuristic.

In [16] a fully time-and-energy-expanded network is proposed, from which a MILP is derived, where the frequency at which the passenger lines are serviced and the number of chargers are decision variables. The formulation is verified using a commercial MILP solver.

A time-and-energy-expanded network with timelines to track charger slot occupation like in our model is proposed in [3]. Two graphs are obtained by rounding charge states up or down, from which in turn primal and dual bounds can be derived to fuel column generation. A diving heuristic explores the branch-and-bound tree in a depth-first manner to obtain integer solutions.

[24], [7] and [6] are a series of papers which propose a local search to generate a set of candidate vehicle courses that the set partition formulation is solved over. For every candidate vehicle course, the local search has to solve the fixed route vehicle charging problem to determine whether the course is energy-feasible and its cost, so a labeling algorithm and recharge event insertion heuristics are developed. [24] introduces linear spline interpolations of the charge curve ζ from which dominance rules for the labeling method can be derived. [13] extends [7] to consider settings where operators may wish to charge at publicly accessible third-party infrastructure with uncertain availability. The problem is solved by a benders-based branch-and-cut algorithm using a modified version of the labeling algorithm.

Other extensions of [24] are [17] and [10]. [17] develops a label-setting algorithm for the pricing problem of the column generation for the set partition formulation based on recursive functions derived from the linear spline charge curve approximation. This is embedded within a branch-and-price-and-cut framework. [10] extends the model and algorithm from [24] to also include non-linear discharging.

Other papers relying on linear spline approximations of ζ are [36], which presents an adaptive large neighborhood search, [37], which develops a label-setting algorithm considering battery capacity fade, and [38], which also considers capacity fade and develops a MILP with a number of a priori tightening inequalities and runs a commercial solver on it. We assess the previously unknown numerical implications of approximating the charge curve ζ by a linear spline interpolation in [23] and [22].

Lastly, there are a few exact approaches. [26] uses a greedy construction heuristic with backtracking to insert charging events such that arbitrary $\Delta\zeta$ can be considered. [4] proposes a branch-and-check algorithm for factory in-plant electric tow trains. A vehicle schedule whose courses can be made energy-feasible by charging for as much as possible is accepted as the optimal solution, otherwise subtour elimination constraints are introduced to prohibit energy-infeasible courses.

[15] considers an objective function that minimizes the total distance and time spent charging. As such, every recharge event in an optimal solution will only charge for as much as is strictly needed to drive the subsequent trips until the next recharge event or the final depot. One can then use column generation on the set of trip sequences that are energy-feasible without charging and their costs can be derived a priori from the inverse of arbitrary charge curves. It is unclear how this approach can work if the objective does not explicitly minimize the time spent recharging. Operators may not want to strictly minimize charging times due to robustness considerations and active charge management [22]. Nevertheless, generating energy-feasible trip sequences is also a core idea behind our method, which further takes advantage of the easier pricing problem that arises then. Our method could loosely be seen as a generalization of [15] to resource constrained vehicle scheduling with replenishment, multiple depots and vehicle types, and arbitrary linear objective functions. It is also the (to our knowledge) first application of the proximal bundle method [11] to *EBSP*.

4 An Integrated Flow and Set Partition Formulation

In computational experiments, we have found that the set partition formulation for *EBSP* has two undesirable properties, which have also been reported in [15] and we expect should be observable in related problems: For one, there are a large number of very similar columns of negative reduced costs, which cause the master problem to quickly become intractable. Furthermore, the longer the vehicle courses can be and thus have more insertion points for recharge events, the worse the pricing problem performs. In contrast, rounding heuristics to produce integer solutions struggle with the flow formulation. Solutions to the LP-relaxation often collect fractional paths into a single bus, charge its battery, and then fractionally distribute the replenished energy into the network. Vehicle courses derived from a (fractional) path decomposition of the LP solution then often share a single recharge event and it is unclear how to efficiently break this up.

Note that a recent publication [29] shows that electric shortest path with recharging and the corresponding minimum cost flow problem are polynomially solvable if $|K| = 1$, the charge curve ζ is piecewise linear, and every minimum cost subpath in the network is also of minimum energy consumption. However, we believe piecewise linear charge curves to be an inadequate model choice to describe the recharge process [23, 21, 22]. Furthermore, energy-optimal subpaths may not be cost-optimal and vice versa in our application. Some trip to trip arcs involve barely any driving because the bus simply idles at a terminal waiting to service the back direction of the corresponding line. Such a turn has a low energy consumption unlike deadheads that involve proper location changes. But the majority of the operational costs on an arc come from the salary for the driver, so it is possible to have a turn arc that is more expensive than a deadhead arc but that requires less energy.

As we have already alluded to in Section 3, we can make the set partition formulation significantly more tractable by limiting P to those paths that start or end at a depot, where vehicles can be removed from the network, or a facility where consumed resources can be restored. All interior nodes of these paths shall be trips or parking spot timeline nodes and the total resource consumption has to be permissible. In the context of *EBSP*, this means the path can be serviced on a single battery charge. We call these paths *vehicle blocks*. Vehicle courses are then alternating sequences of blocks and restoration or recharge events.

We can then couple the set partition formulation with the flow formulation to have it arrange blocks into vehicle courses by forcing all deadheads contained in active blocks to also be active within the flow. Flow conservation will naturally force deadheads to become active that connect blocks to each other or depots. This yields an integrated flow and set partition formulation for *EBSP*.

We can further completely eliminate all resource related constraints from the formulation by assuming that every block $p \in P$ is serviced by a vehicle that has not consumed any resource yet. Consequently, we have to ensure that all block connections are long enough such that a vehicle's resource state can be fully restored from any initial state. We can enforce this in the timeline model via constraints where an active pull-in onto the timeline prevents pull-outs within the appropriate time window from carrying flow. We then only have to consider the maximum driving range when generating vehicle blocks. This is not a restriction in settings where resource restoration happens in constant time, like regular maintenance or battery swapping, but it does prevent partially recharging a battery for *EBSP* and block connections have to have recharge time windows of length $\zeta^{-1}(1)$.

While this is clearly a major restriction compared to allowing partial charging, we hope that the impact on solution quality is limited by the following observations: On most instances we have encountered so far, recharge events are either a depot charging event with a longer time window, especially in rural settings, or an opportunity charging event with a shorter time window during the turn after a trip. Bus timetables usually tighten their frequency in the morning and late afternoon to evening since the passenger demand is higher during these times. This causes a peak of timetabled trips (see Figure 2) and we need at least as many buses as the largest number of simultaneously scheduled trips as a consequence of Dilworth's theorem. Part of these buses will be idle after the morning peak and they can usually be fully recharged before the afternoon or evening, so the corresponding courses should be mostly unaffected by the restriction.

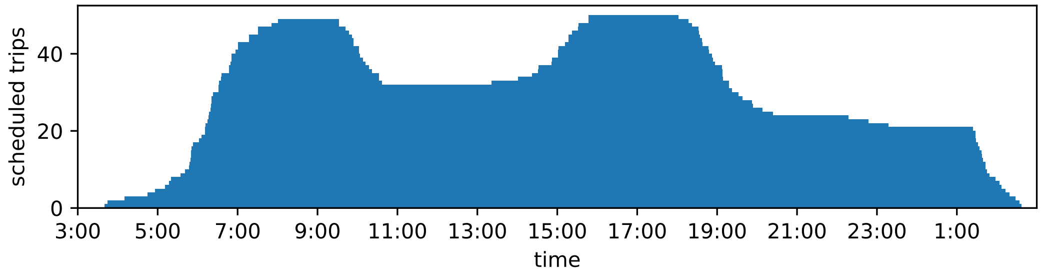

Figure 2 Number of simultaneously scheduled trips of one of our test instances.

Opportunity charging events, as the name suggests, occur during the turns between trips whose terminals are equipped with chargers and a bus can simply recharge there during the mandatory downtime before the next trip. While these events are modeled exactly the same as depot charging via charge slot timelines on the *EBSP* graph, we can efficiently consider them during block generation, unlike depot charging.

Recall that the pricing problem for the full set partition formulation of *EBSP*, where P is the set of energy-feasible vehicle courses, is a resource-constrained shortest path problem with replenishment. This problem is computationally challenging because we can potentially recharge after every trip at any charging facility, for an arbitrary amount of time to an arbitrary state of charge. In particular, the minimum state of charge that the bus has to reach depends on the rest of the path, which in turn depends on how much driving range can actually be restored and the downtime that requires. In fact, as previously mentioned, fitting a cost-optimal recharge schedule to an entirely fixed sequence of trips is NP-hard [24], and the pricing problem to generate vehicle courses is a generalization of this problem.

But if the end terminal of a trip is equipped with an opportunity charger, since the assigned bus is already there, we can simply charge the bus for as long as it can remain depending on whatever trip is put next on the block. Opportunity charging is therefore easy to incorporate into the pricing problem for generating $p \in P$ and we can extend the definition of a vehicle block to allow for opportunity charging whenever a trip terminal is equipped with the necessary infrastructure. This softens our restriction to only apply to depot charging in between blocks, whereas we can consider partial charging for opportunity charging infrastructure as a part of vehicle blocks in P.

We can now give a general *integrated flow and set partition formulation* for *integrated resource constrained vehicle scheduling (with replenishment)* (*IRCVSP*) as

$$
\begin{aligned}
(\mathit{IRCVSP}) \qquad & \min && c^T x && && (21)\\
& \text{s.t.} && M^{\mathrm{F}} x = b && && (22)\\
& && \sum_{p \in P:\, p \ni \tau} w_p = 1 && \forall \tau \in \mathcal{T} && (23)\\
& && \sum_{p \in P_k:\, p \ni a} w_p = x_a^k && \forall a \in A,\ k \in K(a) && (24)\\
& && x_a^k \in \{0,1\} && \forall a \in A,\ k \in K(a) && (25)\\
& && w \in \{0,1\}^P && && (26)
\end{aligned}
$$

where the system (22) is the multi-commodity flow formulation for unconstrained vehicle scheduling as in *BSP*. It enforces the vehicle-mix constraints and that capacities at parking lots and chargers are observed. However, it further needs to guarantee the invariant that vehicles start all blocks with a fully restored resource state by prohibiting block to block connections that are too short. But it is otherwise completely devoid of resource constraints.

Constraints (23) ensure that every trip is covered by exactly one vehicle block and (24) couple the selection of active blocks to that of the active deadheads. Note that if a block contains an opportunity charging event, then that block needs to be coupled to a corresponding charge slot pull-in arc, a number of timeline arcs and a pull-out arc in the flow formulation.

Let $M^{\mathcal{T}}w = \mathbb{1}$ denote (23) and $M^{\text{CF}}x - M^{\text{CP}}w = 0$ denote (24) in matrix notation.

5 Solving *IRCVSP*

Our formulation for *IRCVSP* is superficially similar to standard formulations for the integrated vehicle and duty scheduling problem, where P is defined to be the set of all valid driver duties [34, 2]. A driver duty contains trips and deadheads and has to comply with an underlying vehicle schedule, thus, it can be coupled to a *BSP* flow problem like the vehicle blocks in the *IRCVSP* formulation.

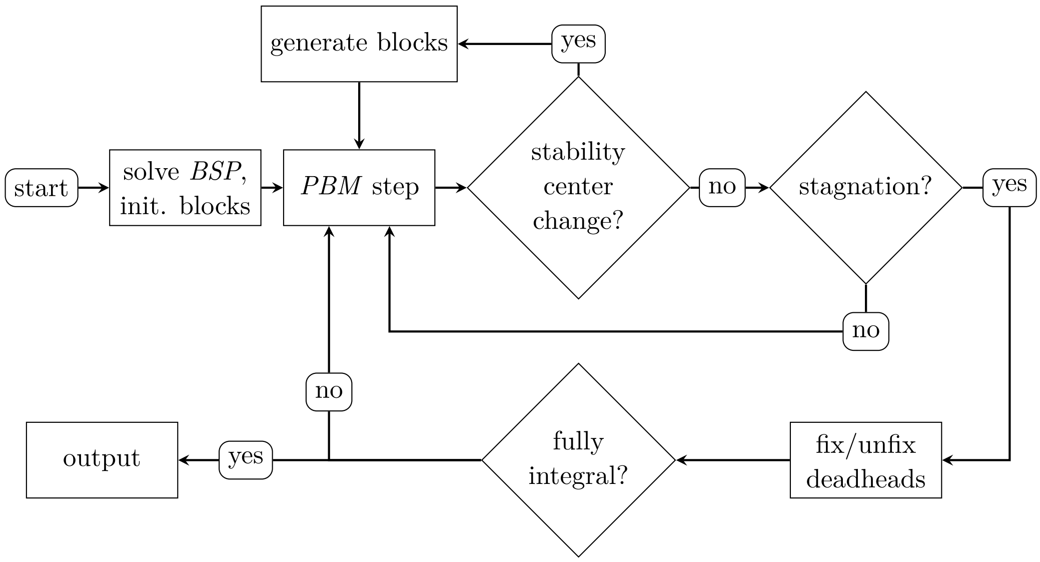

Figure 3 Simplified flow chart of our method to solve *IRCVSP*.

In the case of *EBSP*, there is an intuitive equivalence between drivers and batteries: Both can perform a limited amount of work before they have to be substituted by a fresh replacement. The analogy is literal for battery swapping *EBSP* instances, otherwise replacing a battery by a fresh one means we have to fully recharge it. While driver duties are subject to multiple complicated resource constraints related to working time regulation, we expect similar algorithmic techniques and frameworks used to solve the integrated vehicle and duty scheduling problem to also work for resource constrained vehicle scheduling. In this vein, the algorithm we present here is inspired by our work on the integrated vehicle and duty scheduling problem. Since the algorithmic techniques are quite involved, we only give an outline of the method and describe the necessary modifications to apply it to *EBSP* (see

Figure 3 for a simplified flow chart). For any detailed descriptions omitted here we refer to our previous work [34] and [2]. We further note that the method should generalize to all resource constrained vehicle scheduling problems that fit formulation (*IRCVSP*).

Our method as depicted in Figure 3 has an inner and an outer loop. For the inner loop it relies on the (inexact) *proximal bundle method* (*PBM*) [11] in combination with column generation on the set of vehicle blocks to produce fractional flow values, which are used in a diving heuristic to gradually produce a fully integral flow on the *EBSP* graph by fixing or unfixing arcs as part of the outer loop.

More precisely, we relax the integrality constraints and apply Lagrangian relaxation to the coupling constraints (24) so that the Lagrangian dual

$$\max_{\lambda \in \mathbb{R}^{A \times K(A)}} \Big[\min \quad \left(c^T - \lambda^T M^{\mathrm{CF}}\right) x \quad + \quad \min \quad \lambda^T M^{\mathrm{CP}} w\Big] \tag{27}$$

$$\text{s.t.} \quad M^{\mathrm{F}} x = b \qquad \text{s.t.} \quad M^{\mathcal{T}} w = \mathbb{1} \tag{28}$$

$$x \in [0,1]^{A \times K(A)} \qquad w \in [0,1]^{P} \tag{29}$$

decomposes into the LP-relaxation of the multi-commodity flow and a range-restricted set partition formulation of non-electric *BSP*. The two subproblems are over separate domains and coupled solely via the Lagrange multipliers λ. Therefore, the Lagrangian function (27) is a separable, concave, piecewise linear, and non-smooth function, which can be expressed as $L(\lambda) = f_F(\lambda) + f_P(\lambda)$, where

$$f_F(\lambda) = \min\left\{\left(c^T - \lambda^T M^{\mathrm{CF}}\right) x \,\middle|\, M^{\mathrm{F}} x = b,\ x \in [0,1]^{A \times K(A)}\right\} \tag{30}$$

and

$$f_P(\lambda) = \min\left\{\lambda^T M^{\mathrm{CP}} w \,\middle|\, M^{\mathcal{T}} w = \mathbb{1},\ w \in [0,1]^{P}\right\}. \tag{31}$$

This is exactly the setting for the *PBM* to find the optimal multipliers λ. Given a decomposable concave function such as L, the *PBM* maintains a polyhedral approximation which is iteratively refined along a sequence of so-called stability centers λ_i by evaluating the function components and their subgradients at nearby trial points. Applied to our Lagrangian L from (27), the stability centers λ_i converge towards the optimal multipliers. Furthermore, we can obtain a series that converges towards the optimal primal solution to the LP-relaxation of (*IRCVSP*) from the values x and w that attain f_F and f_P at the trial points [34, 2].

Evaluating f_F and f_P, i.e., solving the LP-relaxations of the flow and the set partition problem for different multipliers, is still computationally challenging and has to be done repeatedly. We therefore solve them approximately, which requires modifications to the *PBM* to still guarantee convergence. For details on this *inexact PBM* for general applications we refer to [11]. How to process approximate evaluations of f_F and f_P is explained in [34, 2].

The flow problem f_F can be (approximately) solved by any appropriate algorithm, we rely on the method described in [18], [19], and [20] as a black-box, which can produce both fractional and integral feasible solutions of high quality as needed. As mentioned before, we have to employ column generation to solve the set partition subproblem f_P, so suppose P^I are the currently selected candidate vehicle blocks from some index set I. If we apply Lagrangian relaxation to this restricted subproblem we obtain

$$\max_{\mu \in \mathbb{R}^{\mathcal{T}}} \left[\mu^T \mathbb{1} + \min_{w_I \in [0,1]^{P^I}} \left(\lambda^T M^{\mathrm{CP}}_{\cdot I} - \mu^T M^{\mathcal{T}}_{\cdot I}\right) w_I\right] \tag{32}$$

where $M_{\cdot I}$ denotes the submatrix made of columns indexed by I. For fixed multipliers λ and μ the minimization is trivial to solve by setting w_i to one if $\lambda^T M^{\text{CP}}_{\cdot\{i\}} \leq \mu^T M^{\mathcal{T}}_{\cdot\{i\}}$ and zero otherwise. For fixed λ from the current *PBM* step of solving (27) we can then use the (exact) *PBM* to determine the optimal μ, which yields an approximation for f_P restricted to P^I and a corresponding argument w_I. It is possible to deduce an approximation of the reduced costs of all vehicle blocks from this by repairing μ^* into a dual feasible, almost optimal solution, see [34] for details. The reduced cost of block i is then $\lambda^T M^{\text{CP}}_{\cdot\{i\}} - (\mu^*)^T M^{\mathcal{T}}_{\cdot\{i\}}$. Since $M^{\mathcal{T}}$ and M^{CP} are simply incidence matrices of which trips and deadheads are contained in which blocks, λ and μ^* yield arc weights which we use for the vehicle block pricing problem as explained in Section 4. It can be solved by standard label-setting techniques. If a deadhead with an opportunity charging window is processed, i.e., a bus can idle at a charger at a trip terminal for a few minutes, we evaluate $\Delta\zeta$ for every respective label. While we eliminate the need to decide when, where and for how long to charge after every trip for the pricing problem and instead offload this decision to the flow subproblem, generating vehicle blocks is still a computationally expensive step, so we only do it when the stability center and thus the candidate trial points of the main *PBM* process changes significantly.

Finally, the lower bounds and (approximated) primal LP-solutions obtained by repeatedly evaluating (27) are used to guide a rounding heuristic to find high quality integer solutions for *IRCVSP*. Once the *PBM* appears to stagnate at the current stability center, we enter the outer loop as indicated in Figure 3 and fix arcs for which x_a is close to 1.0, appropriately propagate this decision through the network and then relaunch the *PBM* algorithm. We dynamically adjust the threshold for when an arc becomes fixed to be more aggressive early on. If a fixing causes the objective to increase by a large margin, the method can backtrack and revert the decision, however, this step is rarely necessary. The final solution is then a feasible binary multi-commodity flow on the vehicle scheduling graph which adheres to all capacity and vehicle-mix constraints, and is straightforward to decompose into individual vehicle courses as explained in Section 2. It is compatible with a selection of energy-feasible blocks which are connected with sufficient downtime to fully recharge the battery, so the vehicle schedule is energy-feasible.

Note that we can obtain an initial solution to start the procedure with by solving the non-electric *BSP* to integrality, then we simply cut the resulting vehicle courses into energy-feasible blocks, which is what the step after "start" in Figure 3 refers to. Throughout the method we also occasionally delete blocks with large reduced cost from P^I to keep the number of candidate blocks tractable.

6 Computational Results

We tested our method on sixteen anonymous real-life *EBSP* instances with sizes depicted in Table 1. Instance G extends F, and I extends H by an additional opportunity fast-charging terminal. Instances K, L and M are variations of the same instance with different charging technology and bus types.

We ran our method until it returned a feasible integral solution and recorded the runtime and objective value. Then, we ran Gurobi 11.0.0 [8] twice on the mixed-integer formulation (7) - (15) for *EBSP* with our charge curve linearization of (13) described in [21] and [22], once with the previously obtained solution and once without any information. After twelve hours of running Gurobi, we recorded the best found objective value, lower bound, and how long it took in the cold-started run to find a solution that was at least as good as the one produced by our method. Note here that our method enforces that the downtime between blocks is large enough to fully recharge an initially empty battery, whereas the MILP permits partial recharge events in the depot. Both permit partial opportunity charging, however. Therefore, the lower bound and best possible objective are in relation to this more flexible charging policy. The results are presented in Table 2 and Figure 4. The experiment was carried out on an AMD EPYC 7542 CPU restricted to two cores and thus four threads per instance.

Table 1 Number of vehicle types, depots, charge slots, timetabled trips and explicitly given deadhead arcs of our test instances. So-called long arcs are given implicitly via pull-in and pull-out arcs at charge slots and parking facilities at the depots.

instance	electric bus types	non-electric bus types	depots	charge slots	trips	deadheads
A	1	0	1	3	121	992
B	1	0	1	2	123	1 078
C	2	0	1	3	146	3 126
D	1	0	1	3	185	1 769
E	1	0	1	8	189	1 977
F	1	0	1	3	232	1 484
G	1	0	1	5	232	2 064
H	1	1	1	6	333	6 363
I	1	1	1	7	333	7 859
J	1	0	1	14	678	15 589
K	1	0	1	43	709	14 431
L	1	0	1	37	709	17 779
M	1	0	1	34	709	21 343
N	2	1	2	12	822	12 390
O	1	1	1	10	837	111 590
P	1	0	1	28	1 207	42 610

Table 2 Runtime comparison between our method and Gurobi on (7) - (15) for *EBSP* without a start solution. The runtime of our method is how long it took to output an integer solution. We compare it to the time Gurobi takes to produce an integer incumbent that is at least as good as the reference solution of our method and the entry of the faster one is highlighted. A value of "-" indicates that Gurobi failed to produce a better solution within twelve hours (43 200 seconds). Gurobi could not produce any feasible integer solution for instances L and P.

runtime (s) of	A	B	C	D	E	F	G	H
our method	**75**	**72**	**413**	165	**196**	371	322	**4 469**
MILP via Gurobi	-	173	4 967	**82**	6 834	**56**	**8**	39 689
	I	J	K	L	M	N	O	P
our method	**6 604**	**1 767**	**1 854**	**5 008**	**2 637**	2 887	29 334	**2 792**
MILP via Gurobi	-	9 929	-	-	-	**297**	**343**	-

Our method can find good solutions faster than Gurobi, as demonstrated on eleven of the sixteen tested instances, with a bias towards the larger ones. On six instances it produces a solution in less than two hours that Gurobi can not beat within twelve. In particular, Gurobi

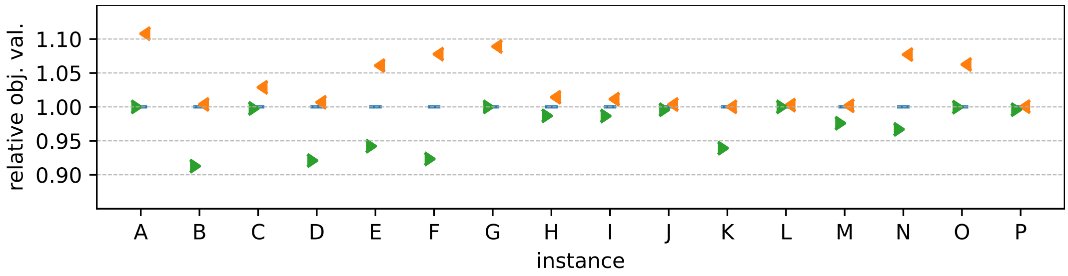

Figure 4 Relative objective values. The best objective value we obtained after letting Gurobi run for twelve hours, both with and without a start solution, is the baseline at 1.0. The best lower bound is given as a relative value by the green triangles pointing to the right. The objective value of the solution produced by our method is given as a relative value by the orange triangles pointing to the left.

fails to produce any feasible vehicle schedule for instances L and P, whereas our method produces solutions that are almost optimal even under the model where partial charging in the depot is allowed, as can be seen in Figure 4.

Integer heuristics used by Gurobi can produce good schedules faster than the overhead of our method permits for the small instances D, F, and G, but those heuristics are not consistent on our test set as can be seen for other small instances A, C, and E. Among the larger instances, N and especially O are the outliers for which our method is significantly slower than Gurobi. Examining the vehicle schedules we see that the problem is the restriction to full recharge windows between blocks. Gurobi can find the optimal solution for O, which admits one hundred and three recharge events, whereas our method produces a solution with sixteen. Most of these recharge events in the optimal solution are short and merely top off an almost fully charged bus, i.e., they are like opportunity recharge events, except the bus takes a small detour of about five minutes to the depot charger. In our data, this depot charger is of course not flagged as opportunity charging infrastructure and is therefore not considered by the vehicle block pricing algorithm. Our method then apparently struggles to produce compatible blocks that allow for full recharge windows in between. We make a similar observation for instance N.

Further examining the objective values and lower bounds in Figure 4, we see that our method solves J, L, and P almost optimally and additionally attains the best found solution for B, K, and M. At worst, it is within 11% of the best found solution and 15% of the best lower bound. Lastly, note that a sequential approach of our method and then Gurobi on the MILP solved A, C, G, J, L, and P to optimality within a total time limit of fourteen hours, whereas Gurobi did not produce any feasible solution for L and P on its own.

References

1 Alan A. Bertossi, Paolo Carraresi, and Giorgio Gallo. On Some Matching Problems Arising in Vehicle Scheduling Models. *Networks*, 17:271–281, 1987.

2 Ralf Borndörfer, Andreas Löbel, and Steffen Weider. A Bundle Method for Integrated Multi-Depot Vehicle and Duty Scheduling in Public Transit. In Mark Hickman, Pitu Mirchandani, and Stefan Voß, editors, *Computer-aided Systems in Public Transport*, volume 600, pages 3–24, 2008.

3 Marelot H. de Vos, Rolf N. van Lieshout, and Twan Dollevoet. Electric Vehicle Scheduling in Public Transit with Capacitated Charging Stations. *Transportation Science*, 2023:1–16, 2023. doi:10.1287/trsc.2022.0253.

4 Heiko Diefenbach, Simon Emde, and Christoph H. Glock. Multi-depot electric vehicle scheduling in in-plant production logistics considering non-linear charging models. *European Journal of Operational Research*, 306(2):828–848, 2023. doi:10.1016/j.ejor.2022.06.050.

5 Tomislav Erdelić and Tonči Carić. A Survey on the Electric Vehicle Routing Problem: Variants and Solution Approaches. *Journal of Advanced Transportation*, 2019:1–48, May 2019. doi:10.1155/2019/5075671.

6 Aurélien Froger, Ola Jabali, Jorge E. Mendoza, and Gilbert Laporte. The Electric Vehicle Routing Problem with Capacitated Charging Stations. *Transportation Science*, 56(2):460–482, 2022. doi:10.1287/trsc.2021.1111.

7 Aurélien Froger, Jorge E. Mendoza, Ola Jabali, and Gilbert Laporte. Improved formulations and algorithmic components for the electric vehicle routing problem with nonlinear charging functions. *Computers & Operations Research*, 104:256–294, 2019. doi:10.1016/j.cor.2018.12.013.

8 Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2024. URL: https://www.gurobi.com.

9 Dominic Jefferies and Dietmar Göhlich. A Comprehensive TCO Evaluation Method for Electric Bus Systems Based on Discrete-Event Simulation Including Bus Scheduling and Charging Infrastructure Optimisation. *World Electric Vehicle Journal*, 11, August 2020. doi:10.3390/wevj11030056.

10 Yong Jun Kim and Byung Do Chung. Energy consumption optimization for the electric vehicle routing problem with state-of-charge-dependent discharging rates. *Journal of Cleaner Production*, 385:135703, 2023. doi:10.1016/j.jclepro.2022.135703.

11 Krzysztof C. Kiwiel. A Proximal Bundle Method with Approximate Subgradient Linearizations. *SIAM Journal on Optimization*, 16(4):1007–1023, 2006. doi:10.1137/040603929.

12 Natalia Kliewer, Taïeb Mellouli, and Leena Suhl. A time–space network based exact optimization model for multi-depot bus scheduling. *European Journal of Operational Research*, 175(3):1616–1627, 2006. doi:10.1016/j.ejor.2005.02.030.

13 Nicholas D. Kullman, Justin C. Goodson, and Jorge E. Mendoza. Electric Vehicle Routing with Public Charging Stations. *Transportation Science*, 55(3):637–659, 2021. doi:10.1287/trsc.2020.1018.

14 Achim Lamatsch. An Approach to Vehicle Scheduling with Depot Capacity Constraints. In Martin Desrochers and Jean-Marc Rousseau, editors, *Computer-Aided Transit Scheduling*, pages 181–195, Berlin, Heidelberg, 1992. Springer Berlin Heidelberg.

15 Chungmok Lee. An exact algorithm for the electric-vehicle routing problem with nonlinear charging time. *Journal of the Operational Research Society*, 72(7):1461–1485, 2021. doi:10.1080/01605682.2020.1730250.

16 Lu Li, Hong K. Lo, and Feng Xiao. Mixed bus fleet scheduling under range and refueling constraints. *Transportation Research Part C: Emerging Technologies*, 104:443–462, 2019. doi:10.1016/j.trc.2019.05.009.

17 Yijing Liang, Said Dabia, and Zhixing Luo. The Electric Vehicle Routing Problem with Nonlinear Charging Functions, 2021. arXiv:2108.01273.

18 Andreas Löbel. *Optimal Vehicle Scheduling in Public Transit.* PhD thesis, Technische Universität Berlin, 1997.

19 Andreas Löbel. Vehicle Scheduling in Public Transit and Lagrangean Pricing. *Management Science*, 44(12-1):1637–1649, 1998.

20 Andreas Löbel. Solving Large-Scale Multi-Depot Vehicle Scheduling Problems. *Computer-Aided Transit Scheduling*, pages 193–220, 1999.

21 Fabian Löbel, Ralf Borndörfer, and Steffen Weider. Non-Linear Charge Functions for Electric Vehicle Scheduling with Dynamic Recharge Rates. In Daniele Frigioni and Philine Schiewe, editors, *23rd Symposium on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2023)*, volume 115 of *Open Access Series in Informatics (OASIcs)*, pages 15:1–15:6, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/OASIcs.ATMOS.2023.15.

22 Fabian Löbel, Ralf Borndörfer, and Steffen Weider. Electric Bus Scheduling with Non-Linear Charging, Power Grid Bottlenecks and Dynamic Recharge Rates. Technical report, Zuse-Institute Berlin, Takustraße 7, 14195 Berlin, Germany, 2024. (pre-print, submitted for review). doi:10.48550/arXiv.2407.14446.

23 Fabian Löbel, Ralf Borndörfer, and Steffen Weider. Non-linear Battery Behavior in Electric Vehicle Scheduling Problems. In Guido Voigt, Malte Fliedner, Knut Haase, Wolfgang Brüggemann, Kai Hoberg, and Joern Meissner, editors, *Operations Research Proceedings 2023*, Lecture Notes in Operations Research. Springer Cham, 2024.

24 Alejandro Montoya, Christelle Guéret, Jorge E. Mendoza, and Juan G. Villegas. The electric vehicle routing problem with nonlinear charging function. *Transportation Research Part B: Methodological*, 103:87–110, 2017. Green Urban Transportation. doi:10.1016/j.trb.2017.02.004.

25 Michael Neißendorfer. Kiepe Electric lädt E-Busse mit Bahnstrom besonders schnell und effizient. *Nahverkehrs-praxis*, 6-2023:60–61, 2023. (German).

26 Nils Olsen and Natalia Kliewer. Scheduling Electric Buses in Public Transport: Modeling of the Charging Process and Analysis of Assumptions. *Logistics Research*, 13(4), 2020. doi:10.23773/2020_4.

27 Samuel Pelletier, Ola Jabali, Gilbert Laporte, and Marco Veneroni. Battery degradation and behaviour for electric vehicles: Review and numerical analyses of several models. *Transportation Research Part B: Methodological*, 103:158–187, 2017. doi:10.1016/j.trb.2017.01.020.

28 Shyam S.G. Perumal, Richard M. Lusby, and Jesper Larsen. Electric bus planning & scheduling: A review of related problems and methodologies. *European Journal of Operational Research*, 2022. doi:10.1016/j.ejor.2021.10.058.

29 Haripriya Pulyassary, Kostas Kollias, Aaron Schild, David Shmoys, and Manxi Wu. Network Flow Problems with Electric Vehicles. In Jens Vygen and Jarosław Byrka, editors, *Integer Programming and Combinatorial Optimization*, pages 365–378, Cham, 2024. Springer Nature Switzerland. doi:10.1007/978-3-031-59835-7_27.

30 Steffen Schulze and Jascha Lackner. Mehr saubere Busse im ÖPNV. *Nahverkehrs-praxis*, 6-2023:50–52, 2023. (German).

31 Ryan Sclar, Camron Gorguinpour, Sebastian Castellanos, and Xiangyi Li. Barriers to Adopting Electric Buses. Technical report, World Resource Institute, 10 g street ne, suite 800, Washington, DC 20002, USA, 2019. last accessed June 2024. URL: https://www.sustainable-bus.com/wp-content/uploads/2019/05/barriers-to-adopting-electric-buses.pdf.

32 Michael Sievers and Andreas Laumen. Großes Potenzial für den Einsatz von eBussen. *Nahverkehrs-praxis*, 6-2023:42–73, 2023. (German).

33 M. E. van Kooten Niekerk, J. M. van den Akker, and J. A. Hoogeveen. Scheduling electric vehicles. *Public Transport*, 9:155–176, 2017. doi:10.1007/s12469-017-0164-0.

34 Steffen Weider. *Integration of Vehicle and Duty Scheduling in Public Transport*. PhD thesis, Technische Universität Berlin, 2007.

35 Weitiao Wu, Yue Lin, Ronghui Liu, and Wenzhou Jin. The multi-depot electric vehicle scheduling problem with power grid characteristics. *Transportation Research Part B: Methodological*, 155:322–347, 2022. doi:10.1016/j.trb.2021.11.007.

36 Aijia Zhang, Tiezhu Li, Ran Tu, Changyin Dong, Haibo Chen, Jianbing Gao, and Ye Liu. The Effect of Nonlinear Charging Function and Line Change Constraints on Electric Bus Scheduling. *Promet - Traffic & Transportation*, 33(4):527–538, 2021. doi:10.7307/ptt.v33i4.3730.

37 Le Zhang, Shuaian Wang, and Xiaobo Qu. Optimal electric bus fleet scheduling considering battery degradation and non-linear charging profile. *Transportation Research Part E: Logistics and Transportation Review*, 154:102445, 2021. doi:10.1016/j.tre.2021.102445.

38 Yu Zhou, Qiang Meng, and Ghim Ping Ong. Electric Bus Charging Scheduling for a Single Public Transport Route Considering Nonlinear Charging Profile and Battery Degradation Effect. *Transportation Research Part B: Methodological*, 159:49–75, 2022. doi:10.1016/j.trb.2022.03.002.