applied system innovation

MDPI

Article

Optimizing Bus Driver Scheduling: A Set Covering Approach for Reducing Transportation Costs

Viktor Sándor Árgilán [1] and József Békési [2,*]

[1] Department of Applied Informatics, Juhász Gyula Faculty of Education, University of Szeged, 6725 Szeged, Hungary; viktor.sandor.argilan@szte.hu
[2] Department of Foundations of Computer Science, Institute of Informatics, University of Szeged, 6725 Szeged, Hungary
* Correspondence: bekesi@inf.u-szeged.hu

Abstract

Cutting operational costs is a critical component for transportation agencies. To reduce these costs, agencies must optimize their scheduling. Typically, the total operating costs of transport include vehicle expenses and driver wages. Solving such tasks is complex, and optimal planning is usually broken down into multiple stages. These stages can include vehicle scheduling, driver shift planning, and driver assignment. This paper focuses specifically on developing a near-optimal driver schedule for a specified set of vehicle schedules. It shows how to efficiently assign drivers to predetermined optimal vehicle routes while ensuring compliance with regulatory constraints on driving hours. We address this challenge using a mathematical model based on the set covering problem, building on a framework established perviously. The set covering problem is typically formulated as an integer programming problem, solvable through column generation techniques. Our algorithm combines this method with heuristics, taking into account the practical aspects of the problem. The article also presents a computational analysis of the method using benchmark and real data.

Keywords: integer programming; driver scheduling; set covering; column generation; heuristics

Academic Editor: Abdelkader Sbihi

Received: 16 July 2025
Revised: 13 August 2025
Accepted: 20 August 2025
Published: 25 August 2025

Citation: Árgilán, V.S.; Békési, J. Optimizing Bus Driver Scheduling: A Set Covering Approach for Reducing Transportation Costs. *Appl. Syst. Innov.* **2025**, *8*, 122. https://doi.org/10.3390/asi8050122

1. Introduction

This paper addresses an important part of the operational planning of public transport services: the planning of driver schedules. The solution methods presented apply to public bus transportation.

Driver scheduling is a crucial component of operational planning for public transportation systems. Public transportation service planning can be divided into several phases. Strategic planning, which usually includes the design of bus routes, is the initial step. Tactical planning encompasses the second and third phases, which focus on establishing service frequencies and schedules, typically determined by the local council. Operational planning includes the final phases, such as arranging drivers, vehicles, and rostering. For example, you can find a detailed description of these phases, models, etc. in [1]. Excellent reviews covering strategic and tactical phases are provided in [2], and many books and studies also address electric vehicles [3–7].

Reducing operating costs is a major goal for transportation firms, as these costs are mostly driven by the cost of vehicles (for example, fuel costs) and their drivers (for example,

fuel costs and driver wages). Vehicle scheduling, vehicle assignment, driver scheduling, and driver rostering are the four stages of operations planning [8].

Vehicle scheduling creates daily vehicle shifts based on the available fleet, ensuring that all scheduled trips are covered (see [9]). Movements of vehicles without passengers, or deadhead trips, are allowed between two planned trips t_1 and t_2 as long as the travel time for the deadhead trip does not exceed the interval between t_1's arrival and t_2's departure. Between the arrival station of t_1 and the departure station of t_2, these deadhead trips take place. We also consider a pull-out at the beginning of the day and a pull-in at the end of the day to be a deadhead trip, if the corresponding stations are different from the depot. Typically, vehicle schedules produce appropriate schedules that meet the requirements of the timetable, such as departure times and locations of the trips. Several mathematical models for solving vehicle scheduling problems have been developed over recent decades. Today, the most widely used models formulate the problem as an integer multi-commodity network flow problem (see [10–13]). In this model, optimal scheduling can be computed as a solution to a linear integer programming problem. The problem can also be formulated as a set covering or set partitioning problem (see, for example, [14,15]). All models, including single- and multi-depot ones, are reviewed in detail in [16].

The vehicle assignment step assigns specific buses to the schedule to meet specific requirements (for example, the maximum distance without refueling for natural gas buses is specified [17]). After this stage, every vehicle shift is assigned to its own bus.

The driver scheduling phase [18] entails assigning drivers to work shifts that meet daily requirements. These limitations include laws that control driving hours, rest intervals, overall work hours, and required breaks between work phases. For example, the maximum time to drive without breaks and the maximum working time per day are prescribed. The most popular approach to solve this problem is to study set partitioning (see [19]) and its relaxation, the set covering model (see [20]). Typically, the solution can be generated in two ways: either by generating a feasible solution satisfying the constraints [21] and iteratively improving it, or by selecting the best among the large number of feasible solutions generated [22]. Numerous types of algorithms have been employed to tackle the problem. Among these, it is worth mentioning the integer programming models (see [23]), the metaheuristic method based on evolutionary principles [22], or algorithms based on the fuzzy method [24].

Driver rostering assigns drivers to shifts over a set period (e.g., weeks or months), while ensuring compliance with scheduling requirements. This includes managing vacation days and meeting minimum rest requirements, such as Sundays off. In transport, split shifts are common, especially on weekdays, to match peak periods (e.g., morning and afternoon commute peaks) with a "two-hump camel back" demand profile. Methods and applications of staff scheduling and rostering are summarized in [25].

Various software modules exist to support operational planning, either manually or interactively, typically as part of a decision support system.

Several integrated models for public transportation incorporate some of the processes mentioned above (see [1,26–30]). However, these models are typically only relevant to medium-sized problems because of their problem size restrictions, which force them to simplify driving regulations or reduce the overall number of rules. Finding a driving schedule that works well is essential to cut operating expenses. Bodin et al. [10] emphasized the importance of this issue as early as the 1980s. They showed that most operational costs are attributable to driver costs, which are typically higher than the average costs associated with cars, using the example of public transportation in North America. Recently, integrated methods have been used in several forms in various related fields, for example, in the optimization of electric buses and rail transport. In [31], Wang et al. integrated

timetabling with vehicle scheduling for urban rail transit, while in [32], Teng et al. used a similar integration for electric buses. In [33], Mo et al. investigated integrated models for paratransit services.

Our Contribution

In the field of vehicle scheduling, the idea of Kliewer et al. [11], the so-called time–space network (TSN) model was a significant advance in the literature in reducing the computational complexity of the model. Its basic idea is presented in Section 2. The model has shown favorable properties, and it has been incorporated into driver scheduling models [26,28,34]. The model continues to play an important role in the mathematical modeling of transportation problems today [35–38]. We use these ideas in our work.

Similarly to the method presented in [26], we construct a driver schedule for an existing optimal vehicle schedule. This is computed by solving a set partitioning problem using column generation. We need column generation because it is not possible to generate all possible shifts since the number of shifts can be very large. In contrast to [26], we generate the driver shifts for the initial column set using a heuristic that is fully adapted to the existing vehicle scheduling and generates shifts that satisfy all rules. Empirical observations show that heuristics are faster than solving a driver scheduling problem. We generate new shifts using a TSN-based shift generation network, similar to the method presented in [28] for another type of integrated problem. This allows us to generate shifts consisting of an arbitrary number of workpieces, which is consistent with the rules we use. Due to the complexity of the rules, it is necessary to incorporate a special checker in the generation network, enabling the handling of more complex rules in addition to the usual resource-based method. This is also an important difference compared to the methods mentioned above. These rules are also described in Section 3.

In Section 2 we describe the technique for calculating vehicle schedules, required by the driver scheduling module. Section 3 contains the driver scheduling method. Finally, Section 4 contains conclusions. The preliminary results of this article are published in the extended abstract [39].

2. Constructing Vehicle Schedules

We outline the process for constructing vehicle schedules. The first step is to create a hypothetical vehicle schedule, and the second is vehicle assignment, the act of allocating certain vehicles to these schedules. The vehicle assignment module and the vehicle scheduling module, respectively, are responsible for these two stages in our system.

The vehicle scheduling module receives the following inputs:

1. Information about the trips, including the departure and arrival times, the geographical locations of the departure and arrival stations, the type of vehicle needed, running distance, and category (local or regional).
2. Vehicle types and related data, such as fuel type and range, running costs for normal and deadhead trips per km.
3. Extra geographic information, including depot-specific characteristics, such as GPS coordinates, number of parking places, suitability for short- or long-term parking, and eligibility for breaks (the required distances and times between the geographic locations can be given or can be calculated from the GPS coordinates).

Various events, including passenger and deadhead trips, are included in vehicle schedules. The multi-depot vehicle scheduling model uses vehicles from particular depots to produce a daily schedule for the fleet. One predefined limitation is the composition of the fleet, which is the total number of vehicles of each category, such as normal, low-floor, and long vehicles.

The following specifications must be met by the vehicle schedule:

1. Each trip must be assigned to a single vehicle.
2. For vehicle shifts to be practical, they must adhere to the following:
   (a) Begin and conclude at the same depot.
   (b) Assignments must be completed without scheduling conflicts.
3. Every depot's capacity restrictions must be adhered to.
4. Requirements for compatibility between depots and journeys must be met.

The following are the general goals of the vehicle scheduling problem:

1. Minimize the number of vehicles required.
2. Minimize the total distance or duration of deadhead trips.
3. Find an optimal (minimum) weighted sum of these costs.

The total cost of a vehicle's daily operation, which varies based on the type of vehicle, is represented by the cost of each vehicle leaving a depot in our cost model. Every vehicle trip also has a variable cost, which is determined by multiplying the cost per kilometer by the distance traveled. These expenses serve as the basis for the objective function that aims to minimize them.

The vehicle scheduling problem has been the subject of several studies published over the last few decades, using a variety of models (see [1] for a summary and comparison of these models and methodologies). Two typical models are used to describe the multi-depot vehicle scheduling problem (MDVSP): the time–space network model (created by Natalie Kliewer et al. in 2006 [11]) and the connection-based network model (as proposed by Andreas Löbel in 1997 [12]). An integer programming (IP) model is solved by both formulations; however, as shown in [11], the time–space network model has fewer arcs. In this study, we use the latter approach for the daily vehicle scheduling software.

Figure 1 shows an example time–space network. There are five geographic locations; two are depots. These locations are represented by horizontal lines that contain the nodes of the network. The nodes represent the departure and arrival times of the trips from or to that station. Various types of edges appear in the network, representing trips, deadheads, and waiting events. Figure 1 uses different colors to distinguish the types of edges:

- The green edges represent deadhead trips to and from the depots.
- The red edges represent deadhead trips between the two stations.
- The yellow edges represent waiting.
- The black edges represent trips.

Multiple edges may occur between some nodes since there are two depots.

We use a specialized assignment model in our system's vehicle assignment module and formulate the problem using linear programming (LP) [17]. This section also addresses vehicle refueling criteria. The input data for the assignment module consist of geographical location data, particular vehicle attributes, and hypothetical vehicle timetables. (We talk about theoretical vehicle scheduling because there is no specific vehicle, e.g., identified by registration number allocated to a shift.) Vehicle fuel types are also included.

Vehicle schedules with precise vehicle identification are included in the output of the vehicle assignment module. In addition, vehicle shifts include specific vehicle events, such as parking and maintenance. The current state of the vehicle (such as the distance driven without refueling and the time elapsed since the last maintenance) and its specifications (such as its maximum range) are among the limitations of the problem. The vehicle assignment module is described in our previous paper [17].

The input of driver planning comes from the phases of vehicle scheduling and assignment; therefore, vehicle schedules and assignments must be given in advance.

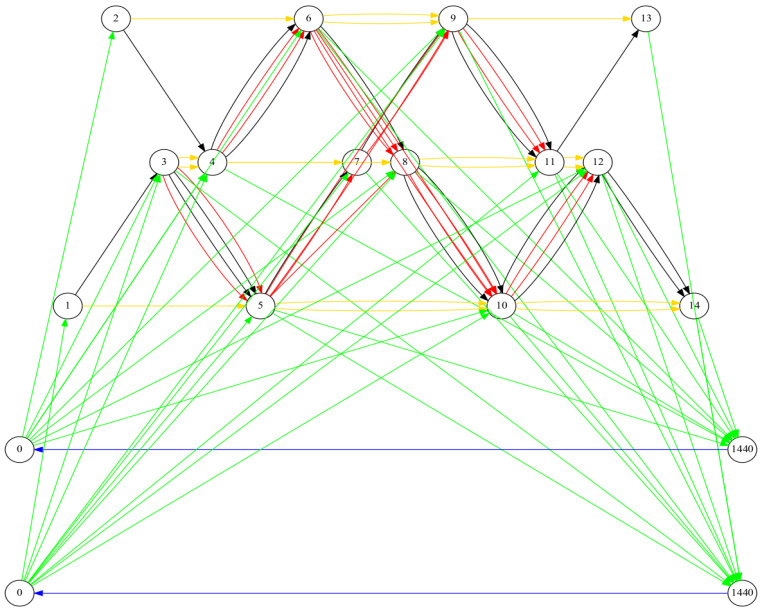

Figure 1. An example time–space network of 12 trips.

3. Constructing Driver Schedules

Now, we will discuss some concepts regarding mandatory driving legislation and driver schedules. We will then discuss a method for scheduling drivers.

Operating guidelines as limitations on driver scheduling

To start with, we define a few keywords and concepts associated with driver employment rules. A valid sequence of trips and deadhead trips is called a "workpiece". "Valid" in this context refers to the possibility of a single driver completing the sequence without interruption. There are some events in driver shifts. For example, events can be special activities that take place at the beginning and end of the shift, which require driving and other events such as shift breaks, vehicle maintenance time intervals, etc. In general, additional rules and regulatory requirements apply to drivers' daily, weekly, and monthly work schedules. However, these parameters are normally covered in the rostering step. Moreover, specific needs must also be considered, such as the type of driver's license required for a certain shift.

Some regulations established by the EU must be considered with regard to the driver schedule, including

1. The maximum number of hours that one can drive in a day;
2. The laws governing the longest period that a driver may spend behind the wheel;
3. The minimum amounts of time needed for rests, breaks, etc.

and local regulations, like the following:

1. Types of the events and shifts;
2. Extra local regulations concerning breaks and the maximum number of working hours (per day).

We will briefly review the set of work guidelines we take into account. First, we present the general rules, established by the EU.

1. A driver cannot exceed 9 h of driving per day, with a maximum of two exceptions per week where it can be extended to 10 h.
2. Breaks of at least 45 min must be taken after every 4.5 h of driving.
3. A break can be divided into two periods, where the first is at least 15 min and the second is at least 30 min (which must be completed after 4.5 h of driving)

The specific local rules for breaks that we must take into account are as follows. These rules mainly apply to working and duty time and not to driving time. Working time may also include other special paid activities such as required waiting, administrative, and maintenance activities.

1. Only specified areas (stations) are allowed for the minimum and maximum duration of breaks, which are 15 and 30 min long, respectively.
2. The driver must take the first work break no later than five hours and fifty-nine minutes into the shift.
3. If the driver works a shift for more than eight hours, a second break must start after working for 8 h and 59 min.
4. If the shift exceeds nine hours, the driver must start a third break before nine hours and 59 min.
5. A shift consists of three periods: the first six hours, the second two hours, and the third one hour. A break may only occur once during a period.
6. The driver is exempt from taking a break if the rest period starts earlier and the shift is finished before the end of the period.

Extra guidelines for operation:

1. If there is no two-hour or longer rest interval during the shift, there must be a minimum 12 h rest period after the shift. The rest period following the shift must be at least 9 h if there is at least a 2 h break during the shift. If this is not the case, the total length of the longest break during the shift plus the rest period following the shift must be at least 12 h.
2. The driver must take a minimum of an 80 min break if their shift is broken into two parts. Short breaks are not allowed during a split shift. A shift must be classified as split if it has a minimum 80 min break. A split shift can have two or more components. In the latter scenario, there must be a minimum of 80 min of rest time between each of the next two shift halves; however, the total of these rest periods cannot exceed five hours.
3. The driver has administrative time at the start and end of a shift. When a shift is split, there is an administrative period at the beginning and end of each part of the shift. They last about five minutes each. A driver change also has an administration period that lasts five minutes.
4. There are certain paid events throughout a driver shift:
    - A half-hour maintenance window for vehicles;
    - A 10 min special administrative period.

    These activities must be offered in blocks of at least five minutes (or in multiples of five).

There are two 2 min breaks for passengers to board and disembark from buses before and after each trip.

In addition, if a shift is shorter than four hours, the full four hours of work must be compensated.

We also implement our approach in regional public transport. The following are the driver regulations for regional public transport:

1. The shifts of the drivers are classified based on the duration of the trips. We refer to a shift as a long shift if it includes a trip of more than 50 km; otherwise, it is referred to as a short shift.
2. After 4.5 h of nonstop driving during long shifts, drivers must take a work break of at least 45 min. There are two possible lengths for this break: the first should be at

least 15 min, and the second should be at least 30 min. Each period may be extended, but the order cannot be altered.

3. A maximum of nine hours may be spent driving on up to three days per week. There is a maximum driving limit of 56 h per week, which doubles to 90 h in two weeks.
4. There is a minimum of 11 h of rest per 24 h. It can be split into two minimal periods: three hours and nine hours.
5. The minimum weekly rest period is 45 h; this can be lowered to 36 h, but the three weeks that follow must make up for the lost time.

Our method to solve the driver scheduling problem

We then present our driver scheduling module's specification and optimization criteria. The input of this module consists of

- Vehicle schedules;
- Vehicle data;
- Driver data (driving license types, contract types);
- Geographical locations.

The driver schedules that are generated must adhere to the established guidelines. It follows that our job is to plan the workpieces and other daily events while adhering to the limitations (daily rules).

In driver scheduling, the cost model is often the total weighted sum of the drivers' fees and the number of shifts. Our objective function aims to minimize the overall cost, which is the sum of the expenses associated with each driver and the number of schedules.

3.1. The Set Covering Formulation Used as the Mathematical Basis for Driver Scheduling

The set cover, or a remarkably similar set partitioning strategy, is the most frequently used mathematical model for driver scheduling in the literature. Numerous articles (e.g., [20,26,28,34]) apply this approach to solve the driver scheduling problem.

We can formulate the driver scheduling problem as a 0–1 programming problem by using the set covering model, specifically,

$$\sum_{d \in D} \sum_{k \in K^d} c_k^d x_k^d \rightarrow \min,$$

$$\sum_{d \in D} \sum_{k \in K^d(t)} x_k^d \geq 1 \text{ for } \forall\, t \in T,$$

$$x_k^d \in \{0,1\},\ d \in D,\ k \in K^d,$$

where:

1. T is the set of timetabled trips to be covered;
2. D is the set of depots;
3. K^d is the set of possible driver shifts from depot d;
4. $K^d(t)$ is the set of driver shifts covering trip t from depot d;
5. c_k^d is the cost of shift k from depot d;
6. x_k^d is the variable indicating if shift k from depot d is included in the solution or not.

The main difficulty when solving the problem with the abovementioned set covering model is that the number of possible combinations of driver shifts is very large. Therefore, they cannot be generated directly. Every driver's shift must adhere to all guidelines and specifications. As a result, we fix the issue using a column generation technique. This approach is widely used in the literature and has numerous uses [40].

The following is a heuristic algorithm that we use to address this issue:

The algorithm of the solution

Step 1 Generate all feasible workpieces from vehicle schedules to form an initial set of columns.
Step 2 Using the current column set K, solve the relaxed master problem. Save the dual program's information as well as the current lower bound.
Step 3 Determine new columns with a negative reduced cost by solving the pricing problem for each depot. Add these new columns to the existing set.
Step 4 Proceed to Step 2 if Step 3 contains new columns with negative reduced costs and the number of steps is less than a specified parameter.
Step 5 Solve the IP problem and build an integer solution with the current column set.

In Step 1 we create the initial driver shifts for column generation using a heuristic that generates all the valid workpieces depending on four parameters from the events of the existing vehicle shifts. These parameters are the minimum and maximum working times and the minimum and maximum number of trips. For each vehicle shift, the method iterates through all the events of the shift and, starting from that event, systematically generates all the valid workpieces beginning with that event. Then with these, we cover each vehicle shift, using the adapted version of the greedy set partition algorithm defined by Chvatal in [41].

The initial driver shifts are made up of all valid workpieces used for this partition. The pseudocode of the algorithms used in Steps 1 is given in Algorithms 1 and 2.

Algorithm 1 Generate workpieces from a vehicle schedule.

```
function GenWP(A, i, MINWT, MAXWT, MINNT, MAXNT)
    ▷ A: Array of vehicle schedules, MINWT, MAXWT: Working time parameters, MINNT,MAXNT: Parameters for the number of trips
    ▷ i: Index of the actual vehicle schedule
    output ← ∅
    ▷ output: Set of the generated workpieces
    E ← A[i]
    ▷ E: Array of events of the ith vehicle schedule
    m ← Length(E)
    for j = 1 to m do
        wT ← 0
        ▷ wT: The accumulated working time in schedule E, starting from the jth event
        nT ← 0
        ▷ nT: The number of trips in schedule E, starting from the jth event
        k ← j
        while wT ≤ MAXWT and nT ≤ MAXNT and k ≤ m do
            wT ← wT + E[k].workTime
            if E[k].type = TRIP then
                nT ← nT + 1
            end if
            if wT ≥ MINWT and nT ≥ MINNT then
                wP ← ∅
                ▷ wP : The array containing the events of the actual workpiece
                for l = j to k do
                    wP.add(E[l])
                end for
                Add wP to the output
            end if
            k ← k + 1
        end while
    end for
    return output
end function
```

Algorithm 2 Solve a set partition problem for each vehicle schedule by a greedy heuristic to obtain the initial driver shifts.

```
function InitialDriverShifts(A, MINWT, MAXWT, MINNT, MAXNT)
    ▷ A: Array of vehicle schedules, MINWT, MAXWT: Working time parameters, MINNT, MAXNT: Parameters for the number of trips
    n ← Length(A)
    output ← ∅
    for i = 1 to n do
        S ← GenWP(A, i, MINWT, MAXWT, MINNT, MAXNT)
        ▷ S: Set of the workpieces generated from the ith vehicle schedule
        while S ≠ ∅ do
            Find s ∈ S containing the most events
            Add s to output
            for all q ∈ S do
                if s ∩ q ≠ ∅ then
                    S ← S − q
                end if
            end for
        end while
    end for
    return output
end function
```

In Step 2 we use LP-relaxation, i.e., we solve the model without the integrality constraints.

In order to select the new columns for each depot, we solve the so-called pricing problem described in Step 3. To do this, we apply the concept of [28] to solve a resource-constrained shortest path problem on a given time–space network *(N,E)* with a given resource set *R*. We use the basic label setting algorithm described in [28].

We considered the additional regulations as well, but resources cannot verify all of the rules. In the pricing problem, additional steps are taken to verify the accuracy of the shifts that are generated using the more intricate drivers' rules. Invalid shifts are excluded and do not appear in the generator. The generator stores several negative reduced cost driver shifts, which will be candidates as the new columns. A parameter determines the maximum number of new columns that can be included in the model.

The structure of the time–space network is similar to that for trips. However, this time the edges of the network are not the trips, but the workpieces. This time, the nodes of the network will be the start and end points of the workpieces on each time line, which are assigned to the end stations. Each generation network also has a depot timeline that represents the current depot. There are four types of edges in the network; these are the pull-in, pull-out, workpiece, and waiting. The pull-in and pull-out edges always start from the depot and lead to the initial node of each workpiece. The workpiece edges are defined by the departure and arrival time and location of the respective workpiece. In the time lines, the workpieces are connected by waiting edges. Theoretically, it is also possible for the waiting to be done by travel, but in our model this type of waiting is not allowed.

In the generator network, certain conditions can be controlled by using resources. Different resources can be assigned to each shift, usually more than one at a time. These express different constraints on the shifts, such as maximum working time, number of work segments included, etc. The resource values for each shift can be calculated using a function. These functions are called resource functions. The driver shifts are described by means of paths; the functions are constructed in such a way that the function value is obtained by summing the values taken at the edges of the path. For more information on resource functions, see, for example, [28,42]. The values of the resource functions for the edges of our time–space network model are given in Table 1.

In our model, the following four resources are used:

1. The maximum working time;
2. The maximum shift duration;
3. The maximum number of workpieces;
4. The maximum driving time.

Table 1. Cost and resource consumption of the different types of edge of the time space network used by the pricing problem. Here, the * symbol means multiplication.

Edge Type	Cost	Resources (Length is Measured in Time)			
		Working time	Shift Duration	Driving Time	Number of Workpieces
Pull-in	Fixed value	Length	Length	Length	0
Pull-out	0	Length	Length	Length	0
Workpiece	Length * Unit cost	Length	Length	Length	1
Waiting	0	0	Length	0	0

Figure 2 shows an example TSN generator corresponding to our model. The notation is the same as in Figure 1, except that here the black edges represent valid workpieces and not trips, and the red edges are missing.

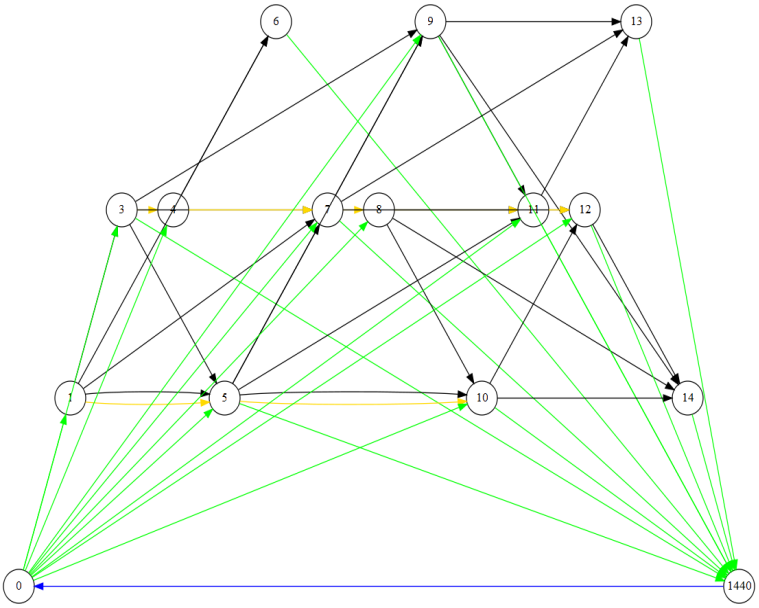

Figure 2. An example of the time–space network-based driver shift generator.

Formally, the pricing problem on a given TSN (N, E) can be stated as

$$\sum_{e \in E} \tilde{c}_e x_e \to \min,$$

supposing that

$$\sum_{e \in n^+} x_e - \sum_{e \in n^-} x_e = \begin{cases} 1, & \text{if } n \text{ is a source node,} \\ 0, & \text{if } n \in N \text{ and } n \text{ is not source and not destination node} \\ -1, & \text{if } n \text{ is a destination node,} \end{cases}, \quad (1)$$

$$a^r \leq \sum_{e \in E} d_e^r x_e \leq f^r, \text{ for each } r \in R, \quad (2)$$

$$x_e \in \{0,1\}, \text{ for each } e \in E.$$

In the objective function, $\tilde{c}_e$ is the reduced cost of edge e, obtained by subtracting the dual value from the original cost value. The binary variable x_e indicates whether the edge e is included in the optimal driver shift. The equations described by Formula (1) express that the driver shift must move continuously from the source to the destination. n^+ is the set of edges that leave node n, while n^- is the set of edges that enter node n. Inequalities (2) imply that regular shifts must satisfy all the criteria defined in the model, e.g., they must not be shorter or longer than a given duration, they must not contain only limited driving time, etc. For each edge, in addition to cost, there are also resource values for each element of the resource set R. These can be, for example, the values defined in Table 1. Here, a^r is the lower bound and f^r is the upper bound for the resource $r \in R$. d_e^r is the corresponding resource consumption for edge e and resource r. The solution of the model gives the regular shift with the lowest reduced cost, which is what we are looking for.

The mentioned resource functions can only be used if the value of the function is obtained as the sum of the values of the events included in the shift. We can compare the values obtained in this way to the appropriate limits. However, as we saw in the introductory part of this chapter, local rules can be special, requiring more complicated conditions to be checked. These include, for example, the different break rules for long and short shifts, or the maintenance and administration periods required for shifts. Such rules can only be checked using a special method. We implement a procedure as part of the shift generator, which checks each incomplete shift to see if it is suitable for placing break events and places the necessary events in the case of already completed shift candidates, thus checking feasibility. Inappropriate shifts are removed from the generator.

3.2. *Computational Results*

The algorithm was first tested on publicly available benchmark data. For this purpose, we selected 20 inputs from the randomly generated data sets discussed in [43,44]. One type of data set, called $A200$, contains 200 trips and 2 depots. The other type, which is called $A400$, has 400 trips in each data series, also served by 2 depots. Huisman et al. in [44] discuss in detail the behavior of their combined vehicle and driver scheduling algorithms developed for these data sets. Tables 2 and 3 show the results of the algorithm described in this paper for these data sets. Although an exact comparison with the results in that paper is not possible, it can be seen that the average number of vehicle schedules is exactly the same, while the number of driver shifts is similar for both input categories. In [44], the average number of driver shifts for the data set $A200$ varies between 40.7 and 44.5, while in our case it is 34.6, and for the data set $A400$ in [44], it varies between 73.2 and 74.9, while in our case it is 62.5. Since the cost function used and the criteria for the driver shifts may differ, the comparison cannot be considered exact, but the results show that for the same data series our algorithm gives similar results to the previous combined methods. As can be seen in the model description, our objective function minimizes the total cost of the driver schedules using optimal vehicle schedules. The objective function used in [44] minimizes the sum of the cost of the vehicle and the driver schedules. Although the exact values of the objective function are not given in [44], this can cause the average number of drivers to be slightly lower when using our model. The number of drivers can also be influenced by the rules, which in our case may also differ from those used in the referenced publication.

The main parameters used in our tests, which correspond to the criteria of the transport company involved in our development, are as follows.

Minimum length of the workpieces: 30 min;

Maximum length of the workpieces: 300 min;
Minimum number of trips per workpiece: 1;
Maximum number of trips per workpiece: not limited;
Maximum number of workpieces per shift: 3;
Maximum working time and shift duration: 720 min;
Maximum driving time: 540 min.

Break rules are checked according to the rules described above.

Table 2. Computational results of the algorithm on random benchmark A200.

Problem	Number of Vehicles	Number of Driver Shifts	Total
#1	19	37	56
#2	21	38	59
#3	18	33	51
#4	17	32	49
#5	20	37	57
#6	16	32	48
#7	18	33	51
#8	20	38	58
#9	23	42	65
#10	12	24	36
Averages	18.4	34.6	53

Table 3. Computational results of the algorithm on random benchmark A400.

Problem	Number of Vehicles	Number of Driver Shifts	Total
#1	34	68	102
#2	35	66	101
#3	34	62	96
#4	29	55	84
#5	35	66	101
#6	29	57	86
#7	32	60	92
#8	36	69	105
#9	43	77	120
#10	22	45	67
Averages	32.9	62.5	95.4

We also use our approach in local and regional public transportation scenarios. The results appear to be encouraging, and the estimated half-hour run times for medium-sized problems are reasonable.

The calculations performed for the Szeged Regional Bus Transport Company are shown in Table 4.

The table shows the number of trips, the number of iterations made by the column generation solver, and the running times. The PC used is configured with an Intel i7-10700 2.90GHz CPU, and 16GB RAM.

In each of the above real-world examples, trips are serviced from three depots, including one larger one (Depot 2) and two smaller ones (Depot 1 and Depot 3). In Tables 5–7, we analyze the behavior of the column generation for these depots. The tables contain the following data:

INS: Initial number of driver shifts, i.e., columns;
FNS: The final number of driver shifts after solving the integer model;
IOV: Initial value of the objective function;

FROV: Final value of the objective function for the relaxed problem;
FIOV: Final value of the objective function for the integer problem;
RG%: Relative gap.

Table 4. Computational results for some real-world problems.

Problem	Number of Trips	Number of Vehicles	Number of Drivers	Number of CG Iterations	Time (minutes)
#1	830	99	225	104	2
#2	902	107	229	143	3
#3	900	105	229	155	3
#4	951	104	201	188	6
#5	1465	175	292	286	30
#6	1467	175	297	272	29
#7	1483	174	298	223	21

Table 5. Number of schedules and values of the objective function for depot 1.

Problem	Depot 1					
	INS	FNS	IOV	FROV	FIOV	RG%
#1	76	58	781,674	603,116	603,116	0
#2	60	36	632,574	360,320.601	377,021	4.42
#3	60	36	632,574	360,320.601	377,021	4.42
#4	69	43	727,607	419,176.834	445,501	5.91
#5	93	49	951,707	496,062.845	507,398	2.23
#6	93	57	973,703	574,549.559	590,071	2.63
#7	90	52	943,671	531,940.470	538,806	1.27

Table 6. Number of schedules and values of the objective function for depot 2.

Problem	Depot 2					
	INS	FNS	IOV	FROV	FIOV	RG%
#1	227	158	2,405,062	1,658,725.778	1,676,210	1.04
#2	190	119	2,032,993	1,228,429.196	1,254,494	2.07
#3	190	122	2,043,245	1,267,100.875	1,285,449	1.42
#4	209	119	2,258,083	1,245,889.563	1,271,081	1.98
#5	322	194	3,450,365	2,014,750.609	2,050,696	1.75
#6	321	192	3,434,485	2,003,882.411	2,038,130	1.68
#7	326	195	3,485,790	2,035,322.634	2,060,903	1.24

Table 7. Number of schedules and values of the objective function for depot 3.

Problem	Depot 3					
	INS	FNS	IOV	FROV	FIOV	RG%
#1	11	9	105,632	95,177	95,177	0
#2	75	74	910,375	895,845	895,845	0
#3	72	71	910,375	851,249	851,249	0
#4	39	39	474,073	474,073	474,073	0
#5	49	49	613,759	613,441	613,441	0
#6	48	48	603,821	603,503	603,503	0
#7	55	51	671,171	627,329	627,329	0

4. Conclusions

In this study, we introduced a novel and application-oriented approach to integrated vehicle and bus driver scheduling using a set covering formulation for the driver scheduling part with an efficient column generation framework and a heuristic for initial shift generation. Unlike previous methods, our approach constructs the initial column set using

a heuristic fully adapted to the given vehicle schedules, ensuring compliance with all operational and regulatory rules. This eliminates the need to generate infeasible shifts and reduces the computational overhead. Furthermore, the method uses a time–space network-based shift generator enhanced with a special feasibility checker, enabling the incorporation of complex, locally specific break and rest regulations that are often oversimplified or ignored in existing models. The originality of this work lies in its combination of three elements: (1) a problem-specific heuristic for initial solution construction, (2) integration of sophisticated regulatory constraints into the column generation process through a dedicated rule checking mechanism, and (3) demonstrated applicability to large-scale, real-world public transportation scenarios with competitive performance against benchmark results. The adaptability and practicality of the method are further validated through the successful deployment in the Szeged public bus transport system, showing that it can bridge the gap between theoretical optimization models and operational decision making.

Author Contributions: Conceptualization, V.S.Á. and J.B.; methodology, J.B.; validation, V.S.Á. and J.B.; formal analysis, J.B.; investigation, V.S.Á. and J.B.; resources; writing—original draft preparation, V.S.Á. and J.B.; writing—review and editing, V.S.Á. and J.B.; supervision, J.B.; project administration, V.S.Á. All authors have read and agreed to the published version of the manuscript.

Funding: The computational tests of this paper were supported by the former Szeged City Bus Company (Tisza Volán). Its new name is Volánbusz. The authors were supported by Grant 7942 of the University of Szeged Open Access Fund.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Desaulniers, G.; Hickman, M.D. Public Transit. In *Handbook in OR & MS*; Barnhart, C., Laporte, G., Eds.; Elsevier: Amsterdam, The Netherlands, 2007; Volume 14. [CrossRef]
2. Ibarra-Rojas, O.J.; Delgado, F.; Giesen, R.; Muñoz, J.C. Planning, operation, and control of bus transport systems: A literature review. *Transp. Res. Part B Methodol.* **2015**, *77*, 38–75. [CrossRef]
3. Ferro, G.; Minciardi, R.; Parodi, L.; Robba, M. Optimization of Electric-Vehicle Charging. Scheduling and Planning Problems. In *Advances in Industrial Control*; Springer: Cham, Switzerland, 2024. [CrossRef]
4. Perumal, S.S.G.; Lusby, R.M.; Larsen, J. Electric bus planning & scheduling: A review of related problems and methodologies. *Eur. J. Oper. Res.* **2022**, *301*, 395–413. [CrossRef]
5. van Kooten, Niekerk, M.E.; van den Akker, J.M.; Hoogeveen, J.A. Scheduling electric vehicles. *Public Transp.* **2017**, *9*, 155–176. [CrossRef]
6. Pasha, J.; Li, B.; Elmi, Z.; Fathollahi-Fard, A.M.; Lau, Y.; Roshani, A.; Kawasaki, T.; Dulebenets, M.A. Electric vehicle scheduling: State of the art, critical challenges, and future research opportunities. *J. Ind. Inf. Integr.* **2024**, *38*, 100561. [CrossRef]
7. Yao, E.; Liu, T.; Lu, T.; Yang, Y. Optimization of electric vehicle scheduling with multiple vehicle types in public transport. *Sustain. Cities Soc.* **2020**, *52*, 101862. [CrossRef]
8. Békési, J.; Brodnik, A.; Pash, D.; Krész, M. An integrated framework for bus logistic management: Case studies. In *Logistik Management*; Physica-Verlag: Heidelberg, Germany, 2009; pp. 389–411.
9. Dávid, B.; Krész, M. Application Oriented Variable Fixing Methods for the Multiple Depot Vehicle Scheduling Problem. *Acta Cybern.* **2013**, *21*, 53–73. [CrossRef]
10. Bodin, L.; Golden, B.; Assad, A.; Ball, M. Routing and Scheduling of Vehicles and Crews: The State of the Art. *Comput. Oper. Res.* **1983**, *10*, 63–211.
11. Kliewer, N.; Mellouli, T.; Suhl, L. A time-space network based exact optimization model for multi-depot bus scheduling. *Eur. J. Oper. Res.* **2006**, *175*, 1616–1627. [CrossRef]
12. Löbel, A. Optimal Vehicle Scheduling in Public Transit. Ph.D. Thesis, Technische Universitaet, Berlin, Germany, 1997.
13. Ercsey, Z.; Kovács, Z. Multicommodity network flow model of a human resource allocation problem considering time periods. *Cent. Eur. J. Oper. Res.* **2024**, *32*, 1041–1059. [CrossRef]
14. Hadjar, A.; Marcotte, O.; Soumis, F. A Branch-and-Cut Algorithm for the Multiple Depot Vehicle Scheduling Problem. *Oper. Res.* **2006**, *54*, 130–149. [CrossRef]

15. Ribeiro, C.C.; Soumis, F. A Column Generation Approach to the Multiple-Depot Vehicle Scheduling Problem. *Oper. Res.* **1994**, *42*, 41–52. [CrossRef]
16. Bunte, S.; Kliewer, N. An overview on vehicle scheduling models. *J. Public Transp.* **2009**, *1*, 299–317. [CrossRef]
17. Balogh, J.; Békési, J.; Galambos, G.; Krész, M. Model and Algorithm for a Vehicle Scheduling Problem with Refueling. In Proceedings of the 9th Workshop on Models and Algorithms for Planning and Scheduling Problems, Abbey Rolduc, The Netherlands, 29 June–3 July 2009; pp. 229–231.
18. Tóth, A.; Krész, M. An efficient solution approach for real-world driver scheduling problems in urban bus transportation. *Cent. Eur. J. Oper. Res.* **2013**, *21* (Supp. S1), 75–94. [CrossRef]
19. Falkner, J.C.; Ryan, D.M. EXPRESS: Set partitioning for bus crew scheduling in Christchurch. In *Computer-Aided Transit Scheduling*; Desrochers, M., Rousseau, J.-M., Eds.; Lecture Notes in Economics and Mathematical Systems; Springer: Berlin, Germany, 1992; Voiume 386, pp. 359–378.
20. Smith, B.M.; Wren, A. A bus crew scheduling system using a set covering formulation. *Transp. Res.* **1988**, *22*, 97–108. [CrossRef]
21. Dias, T.G.; de Sousa, J.P.; Cunha, J.F. Genetic algorithms for the bus driver scheduling problem: A case study. *J. Oper. Res. Soc.* **2002**, *53*, 324–335. [CrossRef]
22. Kwan, R.S.K.; Kwan, A.S.K.; Wren, A.S.K. Evolutionary Driver Scheduling with Relief Chains. *Evol. Comput.* **2001**, *9*, 445–460. [CrossRef]
23. Wren, A.; Fores, S.; Kwan, A.S.K.; Kwan, R.S.K.; Parker, M.E.; Proll, L. A flexible system for scheduling drivers. *J. Sched.* **2003**, *6*, 437–455. [CrossRef]
24. Li, J. A Self-Adjusting Algorithm for Driver Scheduling. *J. Heuristics* **2005**, *11*, 351–367. [CrossRef]
25. Ernst, A.T.; Jiang, H.; Krishnamoorthy, M.; Sier, D. Staff scheduling and rostering: A review of applications, methods and models. *Eur. J. Oper. Res.* **2004**, *153*, 3–27. [CrossRef]
26. Gintner, V.; Kliewer, N.; Suhl, L. A Crew Scheduling Approach for Public Transit Enhanced with Aspects from Vehicle Scheduling. In *Lecture Notes in Economics and Mathematical Systems 600, Computer-Aided Systems in Public Transport*; Hickman, M., Mirchandani, P., Voss, S., Eds.; Springer: Heidelberg, Germany, 2008; pp. 25–42.
27. Mesquita, M.; Moz, M.; Paias, A.; Paixao, J.; Pato, M.; Respício, A. A new model for the integrated vehicle-crew-rostering problem and a computational study on rosters. *J. Sched.* **2011**, *14*, 319–334. [CrossRef]
28. Steinzen, I. Topics in Integrated Vehicle and Crew Scheduling in Public Transit. Ph.D. Thesis, University of Paderborn, Paderborn, Germany, 2007.
29. Weider, S. Integration of Vehicle and Duty Scheduling in Public Transport. Ph.D. Thesis, TU Berlin, Berlin, Germany, 1997.
30. Horváth, M.; Kis, T. Computing strong lower and upper bounds for the integrated multiple-depot vehicle and crew scheduling problem with branch-and-price. *Cent. Eur. J. Oper. Res.* **2019**, *27*, 39–67. [CrossRef]
31. Wang, J.; Yuan, Z.; Cao, Z.; Lu, Z. Optimal Bus Bridging Schedule with Transfer Passenger Demand during Disruptions of Urban Rail Transit. *J. Transp. Eng. Part A Syst.* **2021**, *147*, 04021071. [CrossRef]
32. Teng, J.; Chen, T.; Fan, W. Integrated Approach to Vehicle Scheduling and Bus Timetabling for an Electric Bus Line. *J. Transp. Eng. Part A Syst.* **2020**, *146*, 04019073. [CrossRef]
33. Mo, D.Y.; Lam, H.Y.; Xu, W.; Ho, G.T.S. Design of Flexible Vehicle Scheduling Systems for Sustainable Paratransit Services. *Sustainability* **2020**, *12*, 5594. [CrossRef]
34. Steinzen, I.; Gintner, V.; Suhl, L.; Kliewer, N. A Time-Space Network Approach for the Integrated Vehicle- and Crew-Scheduling Problem with Multiple Depots. *Transp. Sci.* **2010**, *44*, 367–382. [CrossRef]
35. Yin, W.; Hu, W.; Yan, X.; Peng, B.; Yang, X. A time-space network-based model for transportation service optimization of China Railway Express. *High-Speed Railw.* **2024**, *2*, 153–163. [CrossRef]
36. Jiang, Y.; Bian, B.; Zheng, B.; Chu, J. A time space network optimization model for integrated fresh fruit harvest and distribution considering maturity. *Comput. Ind. Eng.* **2024**, *190*, 110029. [CrossRef]
37. Peng, F.; Fan, X.; Wang, P.; Sheng, M. A Time-Space Network-Based Optimization Method for Scheduling Depot Drivers. *Sustainability* **2022**, *14*, 14431. [CrossRef]
38. Qu, Z.; He, S. A Time-Space Network Model Based on a Train Diagram for Predicting and Controlling the Traffic Congestion in a Station Caused by an Emergency. *Symmetry* **2019**, *11*, 780. [CrossRef]
39. Balogh, J.; Békési, J. Driver scheduling for vehicle schedules using a set covering approach: A case study. In Proceedings of the 10th International Conference on Applied Informatics, Eger, Hungary, 30 January–1 February 2014; pp. 219–229. [CrossRef]
40. Desrochers, M.; Soumis, F. A column generation approach to the urban transit crew scheduling problem. *Transp. Sci.* **1989**, *23*, 1–65. [CrossRef]
41. Chvatal, V. A greedy heuristic for the set-covering problem. *Math. Oper. Res.* **1979**, *4*, 233–235. [CrossRef]
42. Irnich, S. Resource extension functions: Properties, inversion, and generalization to segments. *OR Spectr.* **2008**, *30*, 113–148. [CrossRef]

43. Huisman, D. Integrated and Dynamic Vehicle and Crew Scheduling. Ph.D. Thesis, Tinbergen Institute, Erasmus University, Rotterdam, The Netherlands, 2004.
44. Huisman, D.; Freling, R.; Wagelmans, A.P.M. Multiple-Depot Integrated Vehicle and Crew Scheduling. *Transp. Sci.* **2005**, *39*, 491–502. [CrossRef]