Research Article

Andrea Galadíková* and Norbert Adamko

Simulation-based optimization of decision-making process in railway nodes

https://doi.org/10.1515/comp-2024-0014
received January 14, 2024; accepted September 19, 2024

Abstract: Smooth operation of railway stations and yards is vital for the efficient functioning of the whole railway system. Being complex systems, their operation is extremely sensitive to various influences, which makes their management, especially at the operational level, very difficult. Efficient tools to aid the decision-making process of dispatchers of such stations are therefore needed. With an emphasis on increasing the effectiveness of decision support tools, we propose a simulation-based optimization algorithm. This algorithm extracts a dataset from a simulation model and then reduces it to a partial dataset to be able to use specific exact optimization method in operational management. The partial dataset is limited by certain time horizon. The applicability of the proposed algorithm has been verified on two distinct tasks, namely, personnel assignment and service task assignment in a maintenance depot, confirming the usability of the proposed approach.

Keywords: simulation, optimization, decision support, railway

1 Introduction

Throughout its long history, the railway has continued to evolve and develop in all its aspects: technical, operational, human, and organizational. However, the full potential of this mode of transport has not yet been exploited. As in other areas of life, here too it is required to provide a sufficient degree of computerization and automation of many complex processes. It is important to deal mainly with the management of railway transportation nodes to ensure the fluency of all railway transportation. Individual railway transportation nodes are service systems. Trains are the customers and they need to be served using limited resources (e.g., tracks, locomotives, staff). Planning the operation of such complex systems can be problematic, especially because these systems are sensitive to many small changes (disruptions on the track, train delays, lack of staff, etc.) Although planning is an important part of management, it is necessary to deal with operational management as well. The decision-making process in operative management has to be effective and has to be executed in real-time. As decision-making is influenced by a wide range of factors, ensuring support at the operational level using standard methods is challenging due to the size and complexity of the task. For this reason, it is necessary to consider more effective methods of decision-making support in managing the operation of railway transportation nodes.

2 Motivation

Within railway transportation nodes such as marshalling yards or maintenance depots, it is necessary to ensure efficient management at the operational level, which is closely related to the allocation of the limited resources of the railway service system. To keep it running smoothly, these limited resources need to be allocated in an efficient manner. One way to achieve this is using pre-established schedules and plans. In practice, however, this is often not possible due to the interdependencies between different activities within railway node, as well as the impact of random events, such as delays of incoming trains. These situations need to be handled by dispatchers in real time.

Dispatcher's real-time decision-making is a challenging process and would benefit from an adequate decision-support tools. There are currently some systems providing such support [1,2] but, the use of optimization methods is not implemented effectively enough within these systems in practice. In a previous study [3] we have proposed a decision support system for dispatchers

*** Corresponding author: Andrea Galadíková,** Faculty of Management Science and Informatics, University of Žilina, Žilina, Slovak Republic, e-mail: andrea.galadikova@fri.uniza.sk
Norbert Adamko: Simcon, s.r.o., J. Kráľa 19, Žilina 01001, Slovak Republic, e-mail: norbert.adamko@simcon.sk

based on a simulation model. One issue that needs to be addressed to implement such a system is to improve the modeling of decision making in simulation models. In this study, we deal with the design of an algorithm that would use and combine the possibilities of simulation models and optimization methods (exact methods, heuristics). To validate the principle of the proposed algorithm, we decided to demonstrate its use by solving two distinct tasks: allocating drivers within the maintenance depot and addressing the job shop problem at the same location. We hope that the application of the algorithm to two different tasks will confirm its applicability to different types of tasks in a rail transportation node.

Operation of a maintenance depot is a complex process consisting of many interrelated tasks. Trains usually arrive at a maintenance depot because some type of maintenance needs to be carried out on them (cleaning, repairs, etc.). To carry out such maintenance, it is necessary to move the trains between different locations within the yard (e.g., the shed where the train is being cleaned; the assembly pit; the siding). The trains are moved on relatively short distances and the movements of individual trains are carried out by drivers. In our research, we decided to address the minimization of the time of drivers' transfers by foot between the trains. Time required to cover these distances by foot can significantly increase the maintenance time of all trains. Figure 1 shows a simple example of a depot with trains that need to be moved by the drivers between distinct track groups (red rectangles). The request to move Train 2 is issued a few moments before the request to move Train 3. Imagine a driver who just finished moving Train 1 to a service point. After receiving the request to move Train 2, the driver will be assigned to this task and will have to walk back to the waiting track group where Train 2 is located. Then, he/she will move Train 2 to the service track group and after which, he/she can be assigned to move Train 3 back to the waiting track group. The overall time to execute these train movements (and dependent activities) can be significantly reduced by employing better assignment algorithm. In the example case, the driver can wait for the request to move Train 3 to be issued, then move Train 3 back to the waiting track group and first then it would be assigned to Train 2 to move it to the service tracks.

In reality, the dispatcher has the data about the future plans, so he knows that Train 3 will need to be moved in near future and can therefore make an appropriate decision. Unfortunately, simulation models usually do not work with data on future developments, they often use simple approaches such as First In First Out (when selecting individual resources [e.g., drivers], the first resource in the list is selected). Integration of sophisticated methods that include a view into the future would surely improve the quality of such simulation models.

Another problem that is typically found in a maintenance depot is the determination of the sequence of individual maintenance operations to be performed on individual trains. Different trains require different types of maintenances. Type of maintenance depends on train characteristics, e.g., diesel locomotives need refueling, while electric locomotives do not. Trains with a high degree of degradation need more thorough cleaning, replacement of some parts, and so on. Moreover, type of maintenance could change based on when the given maintenance activity was last performed, because checks must be performed at regular intervals, e.g., once a month/year. Examples of such activities are exterior cleaning, interior cleaning, wheel brake inspection, refueling, and many others. Determining the order in which maintenance activities should be performed can be problematic, mainly because certain types of operations can only be performed at certain specific locations within the depot (e.g., above a maintenance pit). Trains arriving at the depot may have different priorities, particularly regarding their scheduled departure time. Therefore, it is sometimes more beneficial to postpone the maintenance of Train 1, which arrived a few minutes earlier, and wait for Train 2, which has its scheduled departure time much earlier than Train 1. If we were to start the maintenance of Train 1 at a specific location, that

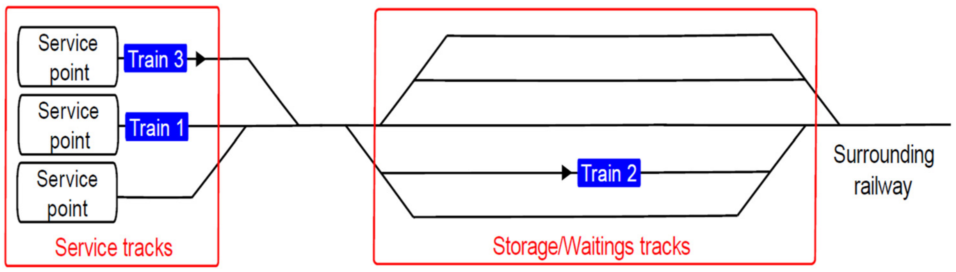

Figure 1: Schema of maintenance depot.

location will be blocked, and the maintenance of Train 2 can only be performed after the maintenance of Train 1 has been completed. This decision can cause undesirable delay of Train 2. In addition, it can happen that the maintenance activity on a train takes a very long time and during this time, several other trains could have been maintained at the specific location. It is then up to the dispatcher to consider whether it is more appropriate to delay one or more trains.

In order to propose a suitable approach to support the decision making process of the railway transportation node dispatcher, an analysis of existing research was carried out.

3 Related work

Several authors have focused on the development of mathematical models and then used exact computational methods to solve these models [4–10]. However, the application of a completely exact approach does not seem to be time-efficient and in many cases proves to be completely infeasible due to the size and complexity of the problems to be solved, since in most cases, we are dealing with nondeterministic polynomial-hard (NP-hard) problems. Therefore, many authors use various techniques that modify the original problem, such as the column generation method [11]. Hanczar and Zandi [12] solved the staffing problem efficiently by generating all work tasks in advance. These tasks are fixed inputs to the mathematical model, and this reduces the computational complexity of a given task. Considering the computational complexity of exact methods, several authors have used heuristics such as genetic algorithms [13–15], tabu search [16–18], or ant colony optimization [19,20]. In addition to these heuristic methods, other authors [21,22] have used the capabilities of neural networks and reinforcement learning algorithms to solve rail transportation problems.

Several authors have researched the simulation-based optimization approach to solve optimization problems in railway transportation [23,24]. Högdahl et al. [25] used a combined simulation and optimization approach to minimize the delay in railway scheduling. First, a mathematical model was constructed to define the conditions to be tracked. Then, a simulation model was created in the Railsys environment, and a solution algorithm was proposed. The proposed algorithm consists of four steps: the selection of the initialization schedule and the definition of a constant for the delays; the execution of a fixed number of simulation runs; the setting of the parameters of the mathematical model based on the simulation results; and the solution of the mathematical model. Experiments have shown that the proposed solution is able to reduce train delays by up to 50%. Licciardello et al. [26] combined the capabilities of microsimulation and optimization algorithms. The proposed solution was based on the continuous communication between the simulation model and the optimization method. In order to find optimal solutions, an optimization module was created that included a heuristic local search algorithm that searches the neighborhood by swapping the order of individual operations assigned to a resource. In the proposed system, the simulation model and the optimization module first exchanged static data about the yard and its resources and then, during simulation run, continuously communicated using predefined messages.

Based on the conducted research, it appears that the combination of computer simulation and optimization algorithms is effective in solving decision-making problems related to the allocation of limited personnel resources or setting priorities for individual service tasks in railway transport.

4 Simulation-based optimization algorithm

Mathematical models can be used to perform exact calculations and solve optimization problems found in railway transportation nodes. However, for real-time decision support in operations management, exact approaches are in most cases too time-consuming and therefore unsuitable.

To overcome this issue, we propose an algorithm that uses a simulation-based optimization approach. This approach aims to reduce the state space and thus the computation time by limiting the number of future tasks to be considered by a time boundary (called a horizon). To execute the proposed algorithm, it is necessary to perform an initialization simulation run. This run is completed before the optimization starts and uses a simple algorithm to allocate each resource. The algorithm is standard in simulation models of this type and is usually based on a priority list implementation. During this simulation run, an initialization dataset is created. Data collection within the initialization simulation run was performed in two parallel phases. First phase collects data about the assignment of resources. Whenever a decision is made to allocate a resource, data about this are written in the output dataset. The second phase is the collection of data about performed activities. As soon as an activity is executed in the simulation model, the data are stored in the dataset. This process is shown in Figure 2.

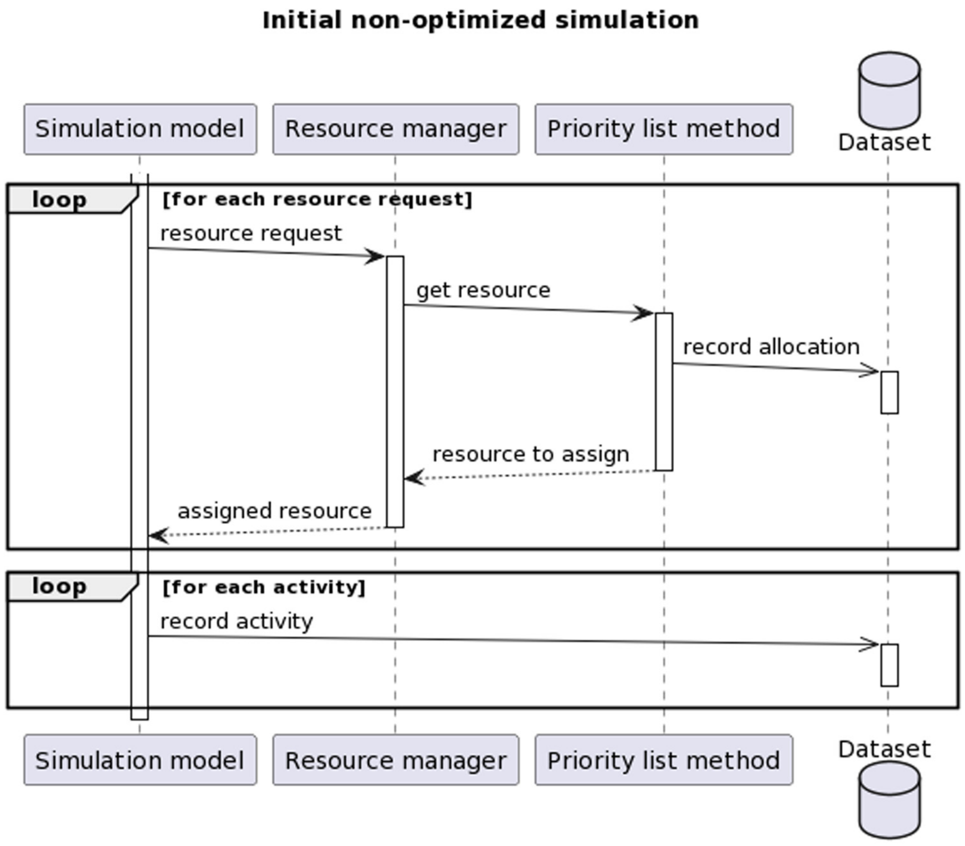

Figure 2: Sequence diagram of simulation-based optimization – initial simulation.

The next step is to launch a simulation run that uses an optimization method. This run differs from the initialization simulation run in the way resource requests are handled – instead of using the priority list method, an optimization method is used and the dataset is used as a source of the necessary information. During this procedure, each request to perform an action, such as allocating a resource (driver) to a train, will be handled in the following way: First, a “get resource” message containing the request is sent to the optimization module. When the optimization module receives this message, it creates a sub-dataset from the initialization dataset. The size of the sub-dataset is determined by the selected time horizon. For example, if the time horizon is 3 h, the sub-dataset consists of all service activities that occur within 3 h of the request. In addition, it only contains information about drivers who are working during this period; it does not include drivers who are currently off duty (due to their work schedule). This significantly reduces the size of the dataset and the computation time of the algorithms used. The optimization module then performs a calculation based on this sub-dataset, determines the most appropriate resource, and sends a message with this resource to the simulation. In the simulation model, the resource is assigned to a request, the request is considered satisfied, and the simulation run continues until the next resource request is encountered. This cycle repeats until the simulation run is complete. Figure 3 shows a sequence diagram of the request processing in the proposed algorithm.

We have decided to verify the proposed approach on two different problems in the maintenance depot, in particular, the personnel assignment problem and the job shop problem. Based on the analysis of exact approaches that have been implemented on similar types of problems in railway, two mathematical models have been proposed for the description of the problems.

5 Mathematical model of personnel assignment

The following mathematical model represents the problem of assigning drivers to individual trains within a maintenance yard as described in Section 2. The mathematical model is inspired by the well-known model of the vehicle routing problem [27], which is modified for the purpose of solving the driver assignment problem addressed in this

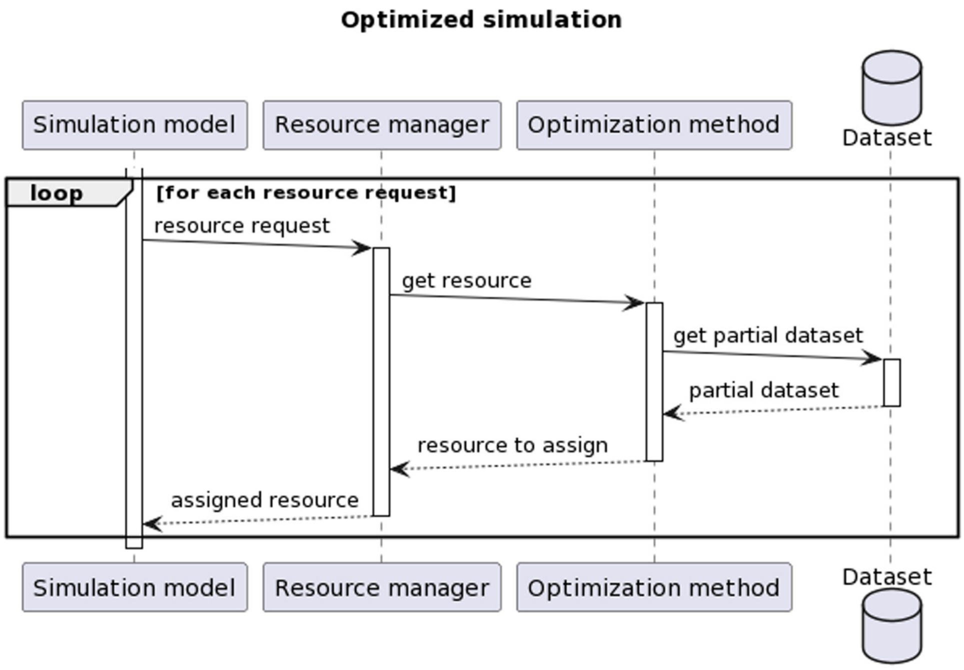

Figure 3: Sequence diagram of simulation-based optimization, using optimization method.

study. In the model description, individual train movements from point A to point B are called transfer activities.

Several sets of variables and constants had to be defined in this model.

x_{ijr} – binary decision variable deciding whether transfer activity i will be executed before transfer activity j by driver r.

y_i – variable, which can take positive values and represents the realistic start time of transfer activity i.

$B(i)$ – a list of transfer activity that can be performed before transfer activity i.

$F(i)$ – a list of transfer activity that can follow after transfer activity i.

t_{ip} – constant that represents the time of arrival of train into the system, i.e., the earliest possible time to start transfer activity i.

t_{io} – constant that represents the duration of the execution of transfer activity i.

t_{rz} – constant defining the start of the working time of driver r.

t_{rk} – constant defining the end of the working time of driver r.

d_{ij} – constant that represents the distance between the point where the driver leaves train on which the transfer activity i is performed and the point where he gets on train on which the transfer activity j is performed.

M – constant defining very large number, used to relax the condition.

N – number of transfer activities of trains.

R – number of drivers.

5.1 Objective function

The objective function (1) consists of two parts. The first part minimizes the distances that drivers must travel between trains they serve (moving within a railway yard). The second part minimizes the time at which train transfers are initiated.

$$\min \sum_{r=1}^{R}\sum_{i=1}^{N}\sum_{j in F(i)} d_{ij}x_{ijr} + \sum_{i=1}^{N} y_i\,. \tag{1}$$

5.2 Structural conditions

Condition (2) expresses that each transfer will be carried out by a single driver. The continuity of the sequence of the individual activities is ensured in condition (3). This condition expresses the fact that some activities (defined by the set B(i)) will not be executed after the execution of activity i. Condition (4) expresses the fact that each driver can only operate one train at a time. Condition (5) expresses that the start of the activity i cannot be executed before the train on which this activity is performed arrives in the system. Condition (6) expresses the fact that if service activities i and j follow each other and they are performed by the same driver, the start of activity j cannot be before the end of activity i plus the time it takes to move between trains belonging to activities i and j in the case of two different trains. If service activities i and j do not follow each other or they are performed by different drivers ($x_{ijr} = 0$), condition (6) is relaxed, and this is ensured by the constant N. Conditions (7) and (8) guarantee that drivers' working hours are respected.

$$\sum_{r=1}^{R} \sum_{i \in B(j)} x_{ijr} = 1 \quad \text{for } j = 2 \dots N, \tag{2}$$

$$\sum_{i \in B(j)} x_{ijr} = \sum_{k \in F(j)} x_{jkr} \quad \text{for } j = 1 \dots N, r = 1 \dots R, \tag{3}$$

$$\sum_{j \in F(1)} x_{1jr} \leq 1 \quad \text{for } r = 1 \dots R, \tag{4}$$

$$t_i^p \leq y_i \quad \text{for } i = 2 \dots N, \tag{5}$$

$$(y_i + t_i^o + d_{ij}) \leq y_j + M(1 - x_{ijr}) \quad \text{for } i = 2 \dots N, \; j \in F(i), \; r = 1 \dots R, \tag{6}$$

$$d_{1j} + t_r^z \leq y_j + M(1 - -x_{1jr}) \quad \text{for } j = 2 \dots N, r = 1 \dots R, \tag{7}$$

$$t_r^k + M(1 - -x_{i1r}) \geq y_i \quad \text{for } i \in B(1), \; r = 1 \dots R. \tag{8}$$

5.3 Obligatory conditions

Constraint (9) defines the range of values that the variables y_i can take, and constraint (10) expresses the binarity of the decision variables x_{ijr}.

$$y_i \geq 0 \quad \text{for } i = 1 \dots N, \tag{9}$$

$$x_{ijr} \in \{0, 1\} \quad \text{for } i = 1 \dots N, j = 1 \dots N, r = 1 \dots R. \tag{10}$$

The following section describes the mathematical model for the second task selected to demonstrate the applicability of the proposed approach, namely, the assignment of the next service task in the maintenance depot.

6 Mathematical model of assignment of service tasks

The following model describes the task of determining the order of maintenance tasks in a maintenance facility, taking its inspiration from the well-known knapsack problem.

Several sets of variables and constants were defined in this model.

x_{ij} – binary decision variable deciding if task i is realized before task j.

y_i – variable, which can take positive values and represents the realistic start time of task i.

v_i – binary decision variable that controls whether a task i is to be executed or not. It is needed, because we are using a time horizon and not every task will be executed.

z_{ij} – binary decision variable deciding if task i is realized in place j.

$C(t)$ – a list of task on train t.

$P(p)$ – a list of tasks, which could be executed on place p.

d_i – constant representing the duration of task i.

a_t – arrival time of train t, the earliest point in time at which any task can begin to be performed on train t.

b_p – the earliest time for start task on place p.

c_i – coefficient expressing the priority of the task i, depending on the departure time of the train, on which task i is performed.

l – current time horizon.

M – constant defining very large number, used to relax the condition.

P – number of places.

T – number of tasks.

6.1 Objective function

The objective function (11) minimizes the time of start of all tasks depending on coefficient c. This coefficient expresses the priority of the tasks. This priority depends on the departure time of the train.

$$\min \sum_{i=1}^{N} c_i y_i. \tag{11}$$

6.2 Structural conditions

Condition (12) expresses that the start of task i cannot be before the arrival of train t at the depot (if this task is to be performed on this train). Since this optimization is based on the partial dataset limited by the time horizon, condition (13) expresses that the start of task i must be before the end of the time horizon if task i is to be executed (when using this horizon, not all tasks from the partial dataset will be performed during the time interval created by the time horizon). Conditions (14) and (15) express that the sum of start time of task i and the duration of task i must be before start time of task j if both tasks are executed and task i is performed before task j on the same train and *vice* *versa*. Conditions (16) and (17) state that if both tasks are executed at the same location and task *i* is performed before task *j*, then the sum of the start time of task *i* and its duration must be before the start time of task *j*, and *vice versa*. Condition (18) expresses that each task must have an assigned place where it is executed. Condition (19) expresses that if task *i* is executed at location *p*, then the start time of task *i* must be greater than the earliest possible start time of using location *p* (location *p* will be free at the earliest possible start time).

$$y_i \geq a_t \quad \text{for } t = 1 \ldots T,\ i \in C(t), \tag{12}$$

$$y_i \leq l + M(1 - v_i) \quad \text{for } t = 1 \ldots T, i \in C(t), \tag{13}$$

$$y_i + d_i \leq y_j + M(3 - v_i - v_j - x_{ij}) \quad \text{for } t = 1 \ldots T, i \in C(t),\ j \in C(t), \tag{14}$$

$$y_j + d_j \leq y_i + M(2 - v_i - v_j + x_{ij}) \quad \text{for } t = 1 \ldots T, i \in C(t),\ j \in C(t), \tag{15}$$

$$y_i + d_i \leq y_j + M(5 - -v_i - -v_j - -x_{ij} - -z_{ip} - -z_{jp}) \quad \text{for } p = 1 \ldots P,\ i \in P(p),\ j \in P(p), \tag{16}$$

$$y_j + d_j \leq y_i + M(4 - v_i - v_j + x_{ij} - z_{ip} - z_{jp}) \quad \text{for } p = 1 \ldots P,\ i \in P(p),\ j \in P(p), \tag{17}$$

$$\sum_{p \in P} z_{ip} = 1 \quad \text{for } i = 1 \ldots T,\ t \in C(t), \tag{18}$$

$$y_i \geq b_p - M(2 - z_{ip} - v_i) \quad \text{for } i = 1 \ldots T,\ p = 1 \ldots P. \tag{19}$$

6.3 Obligatory conditions

The obligatory conditions define the range of individual variables: constraint (20) defines the range of values that the variables y_i can take, and constraints (21)–(23) express the binarity of the decision variables x_{ij}, v_i, and z_{ip}.

$$y_i \geq 0 \quad \text{for } i = 1 \ldots T, \tag{20}$$

$$x_{ij} \in \{0, 1\} \quad \text{for } i = 1 \ldots T,\ j = 1 \ldots T, \tag{21}$$

$$v_i \in \{0, 1\} \quad \text{for } i = 1 \ldots T, \tag{22}$$

$$z_{ip} \in \{0, 1\} \quad \text{for } i = 1 \ldots T,\ p = 1 \ldots P. \tag{23}$$

7 Case study and experiments

The usefulness of the proposed approach has been the subject of several experiments, which were performed with a simulation model of a maintenance depot created in the Villon simulation software [28]. Villon simulation software supports the creation of high-quality microscopic simulation models (especially in the field of railway transportation) that allow detailed evaluation of the modeled system operation. Villon simulation software also provides a broad range of post-simulation statistics for the evaluation of obtained results. This simulation software is based on the agent-oriented architecture ABAsim [29]. In the Villon simulation software, the modelled system is treated as a service system with trains as customers, served by defined stationary (infrastructure, etc.) and mobile resources (personnel, locomotives, etc.). The service of trains is described by flow charts consisting of technological steps called activities. The assignment of each resource to an activity is typically done using a priority list method.

In order to use the proposed algorithm, especially to obtain the initialization dataset, it was necessary to make some extensions to the Villon simulation software. Required data can be extracted directly from the simulation model using an API designed for the needs of this research. It was also necessary to ensure the integration of the optimization method results into the Villon simulation software. This was achieved by using specific set of API methods that allow an intervention in the execution of individual simulation runs, if required.

The task of personal assignment was addressed using the simulation model depicted in Figure 4. In the simulation model, as well as in the real depot system, internal drivers can be in different states, namely, *available, working, delay, moving, return*, and *off duty*. The state *available* means that the driver is on duty but is not currently performing any activity, i.e., not assigned to a train, not moving between trains, and not returning to the depot. The *working* state means that the driver is currently seated on a train and is moving that train from one place to another. A driver is in a *delay* state when he/she is on a train, but the train is waiting for a route to be set or is waiting for the right time to move. In the *moving* state, the driver is walking to the assigned location of the served train, and in the returning state, the driver is walking from the train back to specified home location.

To obtain data from the simulation model, an initialization simulation run was performed, in which individual drivers were assigned to trains using a *First fit* function (although the Villon simulation software provides several algorithms for assigning personnel). The *First fit* function was used in this model because the algorithm used in the real depot was not known when the model was developed. With this method, the first available driver will always be selected and assigned to the task. If there is no driver in *available* state, the train will wait until one of the drivers becomes available.

The dataset created during the initialization simulation run is composed of individual service activities (train

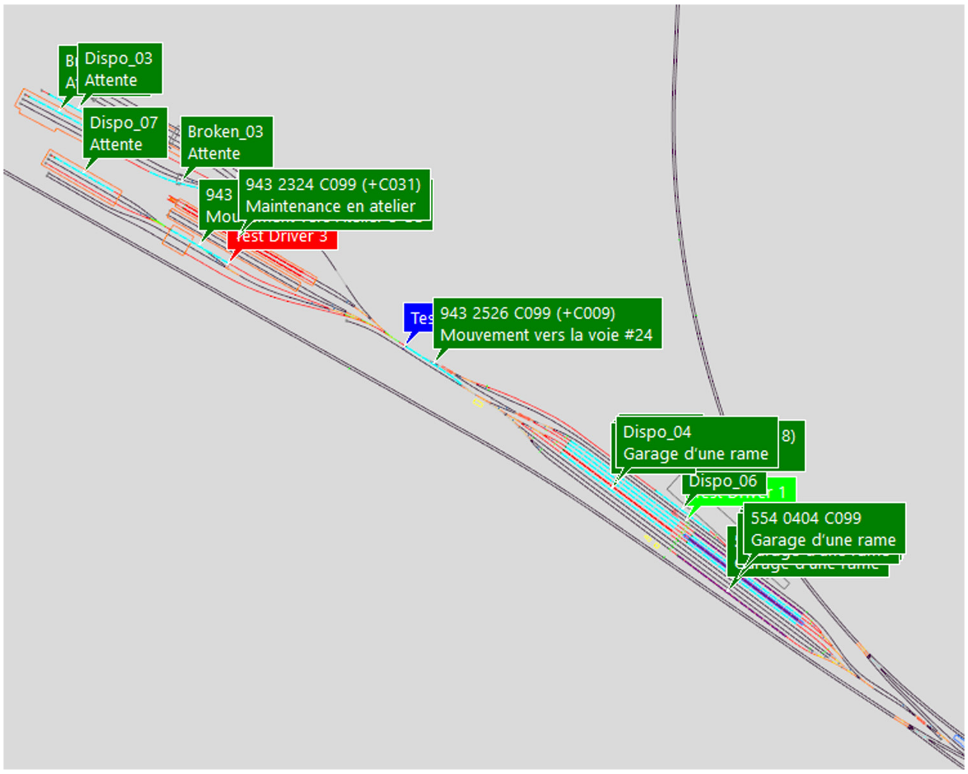

Figure 4: Simulation model 1 in Villon.

movements from point A to point B) and drivers. The individual service activities are always bound to a specific train. For a simulation run of 7 days (7 days of operation in a maintenance depot), a basic dataset of 207 activities and 84 drivers has been created. Note that this does not mean that there are actually 84 drivers, but that there are 84 individual work shifts. It is simpler to describe each shift as a new driver, even though in reality, a driver may have more than one shift in a few days.

The optimization calculations based on mathematical model using this dataset were performed using the standard IP solver Gurobi [30]. The communication between the simulation model and the optimization method was carried out by means of messages in the JSON data format.

The aim of the mathematical model (achieved by minimizing its objective function) is to reduce the duration of the states *moving* and *return* since these are directly related to the distances that the driver has to travel on foot between trains. Reducing the duration of these transfers has a direct impact on the duration of train service as these activities are on the critical path of the train service flowchart. A secondary effect of the optimization should be the increase in the duration of the *available* state and reduction in the duration of the *working* and *delay* states. At the same time, the computational time, in which decision making in depot can be implemented, needs to be maintained. In the framework of our research, the computation time of 1 h is considered a threshold value.

Experiments were carried out with various lengths of time horizon (1, 1.5, 2, 2.5, 3, and 3.5 h) for a 7-day simulation run to investigate the significance of appropriate time horizon settings. The subdatasets included an average of 14 activities and 10 drivers.

Considering that the proposed algorithm should be used to support operative management, it is important to deal with its requirements on computation time. Table 1 shows the computation time for different time horizons. During the execution of experiments, we also tried to conduct an exact calculation on the entire dataset, but the optimal solution was not reached even in 24 h.

Table 2 shows the values of most important states moving and return, and other states were changed insignificantly.

Since the simulation model used to perform the experiments is deterministic, the experiments were performed in a single simulation run. The results obtained using the First fit function (the method originally used) and the results obtained using the proposed approach with a given time horizon were compared by determining by how much the duration of each state was reduced (increased) in percentage.

Table 1: Computation time for different time horizons

Length of the time horizon	Calculation time (min)
First fit	0.3
1 h	0.8
1.5 h	2.1
2 h	3.2
2.5 h	10.5
3 h	15.3
3.5 h	31.2

Table 2: Duration of monitored states for use different time horizons

Length of the time horizon	Moving (h)	Return (h)
First fit	30.1	21.6
1 h	28.2	20.6
1.5 h	28.8	20.4
2 h	28.8	19.7
2.5 h	28.2	19.9
3 h	29.7	19.8
3.5 h	29.0	19.3

Figure 5 shows the improvement of the moving state values when using the proposed simulation-based optimization algorithm compared to the *First fit* function at individual time horizons.

Figure 6 shows the improvement of the *return* state values when using the proposed simulation-based optimization algorithm compared to the *First fit* function at individual changing time horizons.

As it can be seen, the use of the proposed algorithm yielded an improvement over the *First fit* function for both observed states. It may seem that an improvement of

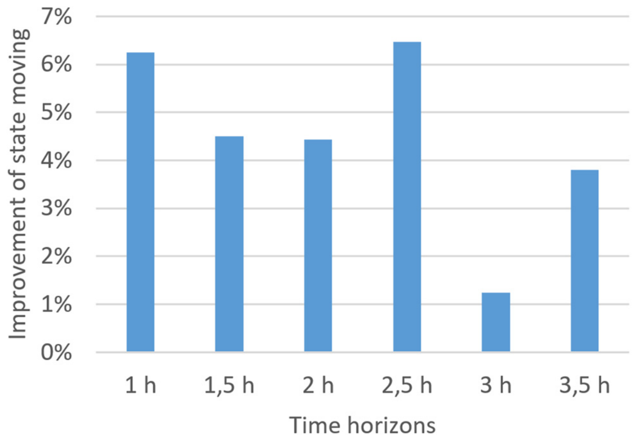

Figure 5: Percentage differences of state moving after using the proposed algorithm with different time horizons.

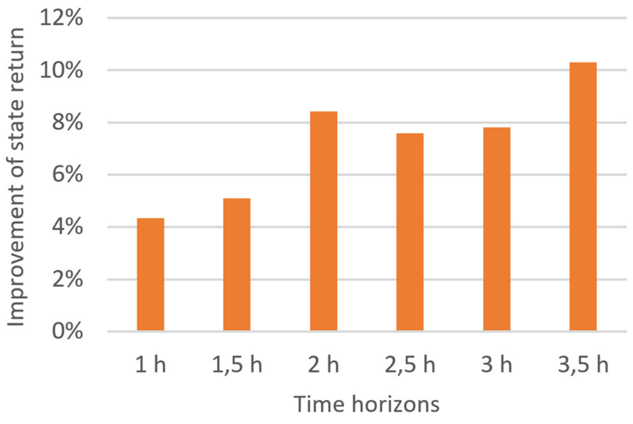

Figure 6: Percentage differences of state return after using the proposed algorithm with different time horizons.

1–6.5% for the moving state and 4–10% for the return state is not significant enough. At this point, however, it is important to mention the limitations for optimizing the given task. The simulation model used is a model of a real maintenance depot. Although the First fit algorithm is used in many places, the individual processes in this model are already partially optimized by the simulation model designer; therefore, the possibilities for further optimization are limited. The correlation between percentage changes of *return* state and the size of the time horizon was confirmed using Pearson's correlation coefficient (the value was 0.9), but no such correlation was found for the *moving* state (value of Pearson's correlation coefficient was −0.56). It is partially possible to conclude that the results improve with increasing time horizon, although the correlation was only confirmed for one of the two monitored indicators (return state). To confirm this conclusion, further experiments with longer time horizons should be carried out. However, experiments with larger time horizons could not be performed because the computation time exceeded the specified threshold. This problem could be eliminated by using heuristic methods instead of an exact approach in the future. Due to the improvement of individual values, it is possible to say that the algorithm improves the results when solving the personnel scheduling problem.

To verify its general usability, the proposed approach of limiting the size of dataset in order to lower the complexity of computation has also been verified on different example model with different optimization problem. This simulation model of maintenance depot is depicted in Figure 7.

In this simulation model, different types of maintenance tasks must be performed on arriving trains, namely, heavy maintenance, light maintenance, interior cleaning, exterior cleaning, graffiti cleaning, and fueling. Each train

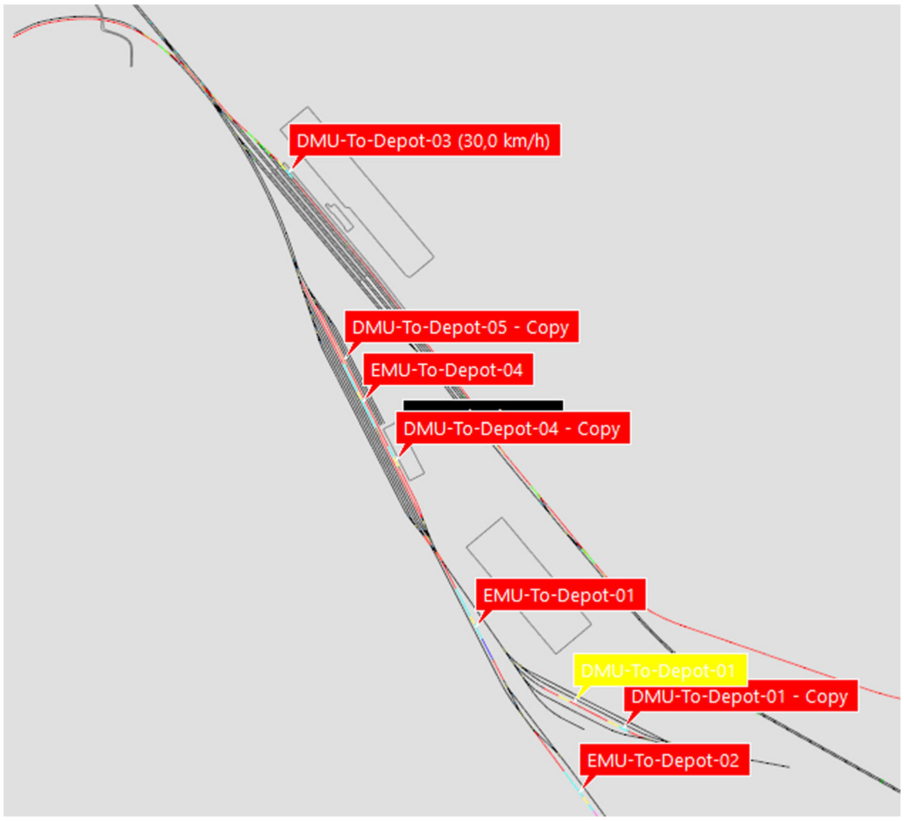

Figure 7: Simulation model 2 in Villon.

requires a specified subset of these maintenance tasks to be executed. The tasks can be performed on specific locations inside the yard – heavy maintenance can be performed on four special tracks; cleaning of interiors, cleaning of exteriors, and cleaning of graffiti can be executed only at one place consisting of two tracks; refueling can be performed on a single track only; and light maintenance can be done everywhere, including parking tracks.

According to the proposed simulation optimization algorithm, the initialization simulation run was completed first. In this simulation run, individual tasks were executed in pre-defined fixed order, given the occupancy of the tracks allowed it. Otherwise, the order was adapted to the current occupancy of the tracks, e.g., if all tracks for heavy maintenance were occupied, the following tasks were executed first (in given order) and heavy maintenance was then performed as the last task for this train. Due to the sufficient number of maintenance tasks in one day, we decided to use a 1-day simulation run for this experiment. The result of the initialization run was the complete dataset consisting of 15 trains and 60 service tasks. Like in the personnel assignment task in previous example model, the IP solver Gurobi was used in this test.

The experiments were carried out on partial datasets, which were created by using different time horizons ranging from 1 to 3.5 h. The partial datasets included in average 7 trains and 30 tasks. It is important to note that the exact calculation works with a partial dataset limited by a specific time horizon; therefore, it is not necessary to include all service tasks.

The goal of this set of experiments was to confirm that the proposed algorithm can determine a more appropriate order of individual service tasks, thereby minimizing train delays. In Table 3, we can see calculation time in different time horizons.

The results of the experiments (values of train delays) are shown in Table 4. The total delay time also includes delays of trains that failed to depart on the respective day. The delay of the unfinished train is calculated as the difference between the scheduled departure time and the simulation finish time.

The percentage change in the sum of delays between the use of the original algorithm and the proposed algorithm with different time horizons is shown in Figure 8.

The positive correlation between the percentage decrease of overall delays and the size of time horizons

Table 3: Duration of calculation in use for different time horizons

Length of the time horizon	Calculation time (min)
First fit	0.2
1 h	0.6
1.5 h	1.2
2 h	5.2
2.5 h	7.4
3 h	20.3
3.5 h	32.3

Table 4: Sum of delays in use for different time horizons

Length of the time horizon	Delay (min)	Number of unfinished trains
First fit	5,234	4
1 h	4,755	4
1.5 h	4,533	4
2 h	4,350	4
2.5 h	4,176	3
3 h	3,120	3
3.5 h	3,445	2

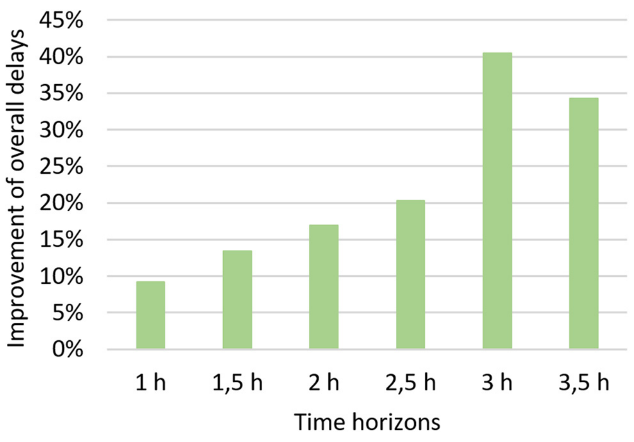

Figure 8: Percentage decrease of overall delays for different time horizons.

was confirmed by calculating the Pearson correlation coefficient, the value of which was 0.78.

8 Conclusion

Designing an effective decision-making method for railway nodes can be difficult, especially considering the complexity of the entire system. Although the exact approach seems to be an appropriate method because it provides an optimal solution, this is not true. If we want to optimize the decisions in decision-making process, it is necessary to consider many factors, and this will cause the task to become larger, and this is the reason why the exact approach does not seem to be suitable due to the time required. To apply the exact approach to larger tasks, we propose a simulation-based optimization algorithm in this work. The proposed approach works with an exact computation that is realized on a partial dataset of the data, which significantly reduces the time required to perform the computations. Although a lot of research has been done in the field of process optimization in the railway, a large part of it does not deal with the possibilities of real-time support and the proposed solutions are very time consuming. In our proposed approach, we perform calculations in the optimization method only on partial datasets. Even though our research could not confirm sufficient time effectiveness for real-time support, we consider the proposed approach valuable because it reduces the computation time requirement in comparison to standard methods. In addition, the related work in the field of railway transport optimization does not adequately take into account the specific problems in the stations, especially in the maintenance depots. These problems are quite different from each other, so methods of solving them could be useful for management of railway.

To confirm the usefulness of the proposed approach, a series of experiments have been carried out. These experiments were conducted on two simulation models of a maintenance depot created in the Villon simulation software. These experiments were related to two distinct tasks, namely, the task of assigning personnel and determining the sequence of service tasks in the maintenance depot. The results of the experiments indicate an improvement over the algorithms originally used for decision making in the simulation models and they lead to the conclusion that the proposed approach is suitable for solving different types of decision-making tasks in a railway transportation node.

We are aware that the proposed approach has only been verified on two types of railway systems, but it is relevant to point out that this verification has been carried out on a complex simulation model containing real data. Since the algorithm is based on standard optimization and simulation methods, we assume that it will be applicable to different types of tasks. For this to be possible, it is of course necessary to carefully consider the use of a suitable optimization method (exact calculation or heuristic) and the availability of a suitable simulation model.

Individual experiments with different time horizons were conducted to confirm the relationship between increasing time horizons and the quality of the results obtained. This relationship was confirmed for two of the three indicators monitored. However, the possibility of increasing the time horizon is considerably limited, since we are trying to provide real-time support in the given approach, and as the time horizon increases, so do the computational requirements.

Even though the proposed algorithm has brought about an improvement, its use in practice is still questionable. Operational management in a railway transportation node must be ensured in real time, and the proposed algorithm does not achieve this speed of calculation. To address this issue, a non-exact (e.g., heuristic) approach could be used for calculations performed on a partial dataset. Further research could be conducted on the use of these methods. Moreover, the applicability in a real node is not yet confirmed and may be problematic because the simulation models used have a considerable degree of abstraction. This should be the subject of further research.

However, the study confirmed the benefits of the proposed simulation-based optimization algorithm, and we believe it could be useful in providing support of decision-making processes for complex railway systems, such as railway transportation nodes.

Acknowledgements: This work is an extension of conference article *Simulation-based optimization of personnel assignment in railway nodes* [31].

Funding information: This work was supported by the research grant VEGA 1/0654/22 *Cost-effective design of combined charging infrastructure and efficient operation of electric vehicles in public transport in sustainable cities and regions and VEGA 1/0077/22 Innovative prediction methods for optimization of public service systems.*

Author contributions: All authors have accepted responsibility for the entire content of this manuscript and consented to its submission to the journal, reviewed all the results and approved the final version of the manuscript. They designed the experiments together and Andrea Galadíková carried them out.

Conflict of interest: Authors state no conflict of interest.

Data availability statement: The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

References

[1] J. Gašparík, D. Lichner, and P. Blaho, Železničná dopravná prevádzka – základy dopravnej prevádzky, Edis, Žilina, no. 1, 2015, p. 407.

[2] Siemens AG, *Mobility Division. Controlguide operations control systems. [Online]*. Available: https://www.mobility.siemens.com/global/en/portfolio/rail/automation/operations-control-systems/controlguide-ocs.html.

[3] A. Galadíková and N. Adamko, "Simulation-based methods to support the real-time management of railways nodes," *Transportation Research Procedia*, vol. 55, pp. 1345–1352, 2021, 14th International Scientific Conference on Sustainable, Modern and Safe Transport.

[4] A. Caprara, M. Fischetti, P. Toth, D. Vigo, and P. Guida, "Algorithms for railway crew management," *Mathematical Programming*, vol. 79, pp. 125–141, Feb 2000.

[5] K. Hoffmann, U. Buscher, J. Neufeld, and F. Tamke, "Solving practical railway crew scheduling problems with attendance rates," *Business and Information Systems Engineering*, vol. 59, pp. 147–159, June 2017.

[6] A. Khosravi Bizhaem and M. a. Tamannaei, "Two mathematical models for railway crew scheduling problem," *The International Journal of Railway Research*, vol. 4, no. 2, pp. 11–22, 2017.

[7] T. Sato, T. Tomiyama, T. Morita, and T. Murata, "Lagrangian relaxation method for network flow modeled crew and vehicle rescheduling," *2010 2nd International Conference on Advanced Computer Control*, vol. 1, 2010, pp. 403–408,

[8] T. Shi and X. S. Zhou, "A mixed integer programming model for optimizing multi-level operations process in railroad yards," *Transportation Research Part B: Methodological*, vol. 80, p. 39, Oct 2015.

[9] A. C. Suyabatmaz and G. Şahin, "Railway crew capacity planning problem with connectivity of schedules," *Transportation Research Part E: Logistics and Transportation Review*, vol. 84, pp. 88–100, 2015.

[10] L. P. Veelenturf, D. Potthoff, D. H. Man, L. G. Kroon, "Railway crew rescheduling with retiming," *Transportation Research Part C: Emerging Technologies*, vol. 20, no. 1, pp. 95–110, 2012, special issue on Optimization in Public Transport+ISTT2011.

[11] J. Janacek, M. Kohani, M. Koniorczyk, and P. Marton, "Optimization of periodic crew schedules with application of column generation method," *Transportation Research Part C: Emerging Technologies*, vol. 83, pp. 165–178, 2017.

[12] P. Hanczar and A. Zandi, "A novel model and solution algorithm to improve crew scheduling in railway transportation: A real world case study," *Computers & Industrial Engineering*, vol. 154, p. 107132, 2021.

[13] E. Khmeleva, A. Hopgood, L. Tipi, and M. Shahidan, "Fuzzy-logic controlled genetic algorithm for the rail-freight crew-scheduling problem," *KI – Künstliche Intelligenz*, vol. 32, pp. 61–75, 2017.

[14] Y. Shen, K. Peng, K. Chen, and J. Li, "Evolutionary crew scheduling with adaptive chromosomes," *Transportation Research Part B: Methodological*, vol. 56, pp. 174–185, 2013.

[15] R. Kwan, A. Wren, and A. Kwan, *Hybrid genetic algorithms for scheduling bus and train drivers*, IEEE, vol. 1, Sep 2000, pp. 285–292,

[16] T. Kokubo and Y. Fukuyama, *Practical train crew scheduling problems using parallel tabu search*, IEEE, Sep 2018, pp. 1673–1678.

[17] K.-L. Chew, J. Pang, Q. Liu, J. Ou, and C. Teo, "An optimization based approach to the train operator scheduling problem at Singapore MRT," *Annals of Operations Research*, vol. 108, pp. 111–122, Nov 2001.

[18] C. Guillermo and M. Jose, *Hybrid algorithm of tabu search and integer programming for the railway crew scheduling problem*, IEEE, vol. 2, Nov. 2009, pp. 413–416.

[19] G. Shi, C. Zhu, and Z. Tian, "Modeling and algorithms of the crew scheduling problem on high-speed railway lines," *Journal of Information and Computational Science*, vol. 12, pp. 2655–2664, July 2015.

[20] W. Zhou, X. Yang, L. Deng, and J. Qin, "Crew scheduling considering both crew duty time difference and cost on urban rail system," *PROMET – Traffic and Transportation*, vol. 28, pp. 449–460, Oct 2016.

[21] A. Bretas, A. Mendes, S. Chalup, M. Jackson, R. Clement, and C. Sanhueza, *Modelling railway traffic management through multi-agent systems and reinforcement learning*, Modelling and simulation society of Australia and New Zealand, 2019, pp. 291–297.

[22] H. Khadilkar, "A scalable reinforcement learning algorithm for scheduling railway lines," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 2, pp. 727–736, 2019.

[23] S. B. Layeb, A. Jaoua, A. Jbira, and Y. Makhlouf, "A simulation-optimization approach for scheduling in stochastic freight transportation," *Computers & Industrial Engineering*, vol. 126, pp. 99–110, 2018.

[24] M. Shakibayifar, A. Sheikholeslami, and F. Corman, "A simulation-based optimization approach to reschedule train traffic in uncertain conditions during disruptions," *Scientia Iranica*, vol. 25, pp. 646–662, Aug 2017.

[25] J. Högdahl, M. Bohlin, and O. Fröidh, "A combined simulation-optimization approach for minimizing travel time and delays in railway timetables," *Transportation Research Part B: Methodological*, vol. 126, pp. 192–212, 2019.

[26] R. Licciardello, N. Adamko, S. Deleplanque, P. Hosteins, R. Liu, P. Pellegrini, et al., "Integrating yards, network and optimization models towards real-time rail freight yard operations," *Ingegneria Ferroviaria*, vol. 75, no. 6, pp. 417–447, 2020.

[27] J. C. Beck, P. Prosser, and E. Selensky, "Vehicle routing and job shop scheduling: What's the difference?," in: *Proceedings of the 13th International Conference on Artificial Intelligence Planning and Scheduling*, 2003.

[28] Simcon. Villon simulation tool. [Online]. Available: http://www.simcon.sk/en/tools/villon.

[29] A. Kavička, V. Klima, and N. Adamko, "Simulations of transportation logistic systems utilising agent-based architecture," *International Journal of Simulation Modelling*, vol. 6, p. 13–24, 2007.

[30] Gurobi. Gurobi optimizer. [Online]. Available: www.gurobi.com.

[31] A. Galadíková, and N. Adamko, "*Simulation-based optimization of personnel assignment in railway nodes*," In: 2022 IEEE 16th International Scientific Conference on Informatics (Informatics). 2022, pp. 1–6.