Received 16 October 2023, accepted 26 November 2023, date of publication 1 December 2023,
date of current version 13 December 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3339128

RESEARCH ARTICLE

Determination of Weights for Multiobjective Combinatorial Optimization in Incident Management With an Evolutionary Algorithm

PHILLIP GACHNANG[1], JOACHIM EHRENTHAL[2], RAINER TELESKO[1], AND THOMAS HANNE[1]
[1]Institute for Information Systems, University of Applied Sciences and Arts Northwestern Switzerland, 4600 Olten, Switzerland
[2]Institute for Information Systems, University of Applied Sciences and Arts Northwestern Switzerland, Windisch, 5210 Brugg, Switzerland

Corresponding author: Thomas Hanne (thomas.hanne@fhnw.ch)

This work was supported by the Innosuisse under Grant 37294.1 IP-ICT.

ABSTRACT Incident management in railway operations includes dealing with complex and multiobjective planning problems with numerous constraints, usually with incomplete information and under time pressure. An incident should be resolved quickly with minor deviations from the original plans and at acceptable costs. The problem formulation usually includes multiple objectives relevant to a railway company and the employees involved in controlling operations. Still, there is little established knowledge and agreement regarding the relative importance of objectives such as expressed by weights. Due to the difficulties in assessing weights in a multiobjective context directly involving decision makers, we elaborate on the autoconfiguration of weighted fitness functions based on nine objectives used in a related Integer Linear Programming (ILP) problem. Our approach proposes an evolutionary algorithm and tests it on real-world railway incident management data. The proposed method outperforms the baseline, where weights are equally distributed. Thus, the algorithm shows the capability to learn weights in future applications based on the priorities of employees solving railway incidents which is not yet possible due to an insufficient availability of real-life data from incident management.

INDEX TERMS Evolutionary algorithm, incident management, multiobjective optimization, railway operations, weight assessment.

I. INTRODUCTION

Railway companies need to run their passenger operations on time and safely. Service interruptions brought about by operational, technical, and infrastructure-related incidents must be resolved quickly and cost-effectively, abiding by guidelines, laws, and regulations. This usually lies in the hands of incident management teams. Depending on the railway company and country-specific regulations, train dispatchers usually handle incidents in collaboration with infrastructure and network managers, and staff on-site, along the respective train routes. Increasingly, railway companies rely on computer-aided dispatching and incident resolution software. However, these systems are static and tend to provide little optimization capabilities in decision support. This is due to operational, technical, and infrastructure complexities in railway management and the immense solution space that dispatchers can call upon to resolve an incident.

Our current study originated from a project together with a provider of railway operations planning software and some client railway companies with the aim to improve incident management processes by better software support. Based on several objective functions specified in cooperation of the project partners, we consider a related aggregative objective function using weights, which can be classified as an integer linear optimization problem.

As the weights turned out to be difficult to determine by decision makers, we consider an approach of automatically determining these weights based on a second-level optimization problem for minimizing the deviation of first-level

The associate editor coordinating the review of this manuscript and approving it for publication was Liangxiu Han.

optimization results from historical planning decisions. This problem could be considered a machine learning problem, but as the underlining model structure is given, general approaches such as, e.g., neural networks appear less suitable.

We find that using an evolutionary algorithm on top of the given weighted optimization function yields excellent results in pre-determining problem-specific weights.

Our results are relevant not only in the domain of railway incident response but in all optimization settings where aggregated objective functions are difficult to define or continuously change regarding priorities or concerning the environment relevant for decision-making. Thus, the main contributions of our study can be summarized as follows:

- provision of a general approach of specifying complex decision models (such as for weight values in a multiobjective decision scenario) using data from previous planning
- proof of concept that an evolutionary algorithm works well for solving such a problem.
- determination of specific parameter values for an objective function in railway incident management

The subsequent paper is structured as follows: In Section II, we discuss related work, on the one hand from the field of railway management, on the other hand regarding the specific problem of weight determination in multiobjective decision scenarios. In Section III, the considered integer linear optimization problem is discussed. The evolutionary algorithm for determining weights is discussed in Section IV together with the respective fitness function and further adaptations for the considered problem. Results of using this approach are presented in Section V. In Section VI the results are discussed, and conclusion are presented in Section VII.

II. RELATED WORK

A. RELATED APPLICATIONS IN RAILWAY MANAGEMENT

The literature for the railway domain often refers to ''incident management'' and ''failures'' in a broader sense. However, we must consider that most research is done for well-defined, specific scenarios that are difficult to generalize. Additionally, based on the available data and data types, e.g., historical reports, image data from video cameras, and simulation data, one must decide what technique is appropriate. This makes a generalization even more difficult.

A comprehensive overview of the application of machine learning (ML) techniques for various railway scenarios discussed in the literature with emphasis on safety, e.g., rail maintenance, track inspection, fatigue crack detection and sizing, classification of accident causes and railway defects, detection of wheel defects, etc. is given in [1]. The ML techniques mostly used in these scenarios comprise decision trees, deep learning, convolutional neural networks, text mining, Bayes networks, and fuzzy models.

A discrete-time train and passenger simulation engine for urban railway networks is introduced in [2]. The problem of finding an optimal action plan (AP) as a response to a localized spatio-temporal incident was modeled with a mixed integer programming formulation. For performance reasons, the simulation engine was implemented in C++ using efficient customized data structures. To check the feasibility of the approach, two real incidents were selected. In incident 1, the service of a line between two stations is disrupted for a longer period of time in both directions because of an infrastructure fault. For this complex situation, several AP are computed, from decreasing the bus shuttle service follow-up times (AP1) to decreasing the train service follow-up times along the entire impacted line or service (AP2), ending up with proposing to create a non-standard train service that serves the surrounding area of concern. Incident 2 happens during the morning peak and leads to a service disruption between two stations lasting for only ten minutes. The computed AP proposed a shuttle bus service with a predefined frequency. The proposed approach is consistent with our research problem, i.e., finding an optimal solution for a well-defined incident, however the solution space is different because other means of transport – besides the urban railway – may be part of a AP.

In [3], a solution for the railway wagon flow routing problem is proposed to improve transportation efficiency. The model is based on the shortest route algorithm of Dijkstra and fully considers factors such as the influence of capacity, the rule of a detour, and the rationality of route selection. Due to some model simplifications and the vast amount of data, a direct implementation has not yet been performed.

In [4], a joint simulation project - with the participation of Alstom - in the area of railway fleet maintenance management is presented together with two real use cases from the United Kingdom. The simulation model aims to visualize the complex interactions in a railway system, i.e., rolling stock, depots, and maintenance guidelines. Based on the collected requirements, such as fleet operation (timetables and availability), maintenance engineering (maintenance regime), and depot management (depot restrictions), the authors developed a heuristic scheduling algorithm that emulates a fleet planner. It is also possible to parameterize the system to represent different railway systems via trainset definitions, the scheduling horizon, etc.

Case-based reasoning (CBR) is becoming increasingly popular in the railway domain. It follows the basic idea of solving new problems based on the solutions of similar past problems and can therefore be used without the specific formulation of the problem as a (single- or multiobjective) optimization problem. In [5], a knowledge-based approach is presented for preventing railway operation incidents consisting of two processes: incident analysis and dispatch advisory. The incident analysis gathers ontological-based information about an incident. The dispatch advisory component is fed with the incident analysis tree to generate appropriate prevention strategies based on CBR or rule-based reasoning if no appropriate case can be found. The incident analysis is processing the domain ontology via various hierarchically ordered components such as the problem type

(PT), problem feature (PF), and possible causes (PC). The PT layer represents the type of the problem, e.g., command problem, judgment problem, etc. PF defines respective problem features for a particular PT. For example, for the PT "command", a PF might be an "incorrect command". The PC layer shows the possible causes which are linked to the PT. PCs for "incorrect command" might be "wrong receiver", "receiver's misunderstanding", or "command too late". Overall, the approach is interesting and relevant to our work, however, the development of a railway domain ontology and the respective (CBR) reasoning is challenging and needs ongoing effort for testing and maintenance.

In [6], an improved CBR method (IMCBR) is described for technical incidents related to turnout systems which are used to control the route direction of a train. The main idea was to use a sophisticated distance metric to get a more accurate retrieval result and to validate that the diagnosis results for the extracted features are reliable. The evaluation of the whole CBR system was carried out with a real data set, proving that the diagnosis performance was enhanced.

In [7], a CBR framework for the automatic train conduction scenario is discussed where the adaptation task consists of a multiobjective optimization approach. The target problem is expressed with the following data: (i) the track the train must travel (initial km, final km; % of incline; and so forth); and (ii) the train composition (e.g. number of locomotives and cars) The case description uses the following attributes: applied/executed action, track topology, initial distance travelled, number of locomotives, number of cars, initial speed in km/h, final speed, maximum speed in km/h, ramp percentage, and total displacement distance. The case memory finally contained 3624 cases. In [7], the 50 most similar cases are used for the adaptation stage, thus establishing the initial population of the genetic algorithm (GA). The problem formulation of the GA considers two objectives, minimizing fuel consumption and minimizing the total travel time. The results with the combined CBR/GA approach are promising when regarding the concrete evaluation results. For some scenarios, the economy of fuel was about 51%, and, in the best-case scenario, a reduction of about 69.5% of travel time was achieved.

In [8], it is suggested that GAs are competitive alternatives for training deep neural networks for reinforcement learning, which this research proposes.

From the existing literature we can conclude, that mostly rather simple or specific incident scenarios are considered, whereas in our case a generic large-scale real-life problem model (based on a commercial railway operations planning software) is considered involving nine objectives (see Section III). In addition, for this problem formulation the determination of weights as priority indicators for the objectives turned out to be a significant problem so that we discuss the weight determination problem in more detail in der next subsection.

B. BACKGROUND ON THE PROBLEM OF DETERMINING WEIGHTS

Challenges and difficulties in determining weights in the context of decision making in general and more specifically for weighting different criteria or objectives (as in multicriteria decision-making, multiattribute decision analysis and multiobjective optimization) have been known for a long time. Some early related works are [9], [10], or [11]. The main points of criticism regarding the usage of weights are as follows [11]:

- The meaning and implications of weights can be ambiguous or misunderstood.
- Issues of commensurability can be overlooked and may lead to contradictory results.
- There may be no setting of weights leading to the most preferred solution of a decision maker.

Although the idea of using weights appears simple and straightforward (or possibly rather because of this), decision makers (DMs) have difficulties in determining them when asked directly. On the one hand, it is difficult to transform their perception of importance (or whatever their interpretation of weights is) into specific weight values. On the other hand, DMs have usually little understanding how a decision approach actually employs the weights. An old approach to mitigate such problems in multicriteria decision making is not to use weights defined apriori (i.e., prior to the use of a multicriteria method) but to involve the decision maker more strongly in the method-based decision processes through interactive approaches [12], [13]. However, also with these methods, problems of methodological nature such as the complexity of a method, the understanding of a method, and other behavioral issues remain [14], [15]. For instance, [16] mention 13 different interpretations of weights as they are used in multicriteria methods including the following: marginal contribution per unit of the objectives, indifference trade-offs or rates of substitution, gradients of the overall value function, scaling factors converting criteria into commensurate overall values, relative contributions of the average criterion specific scores, relative contribution of the criteria to the optimal alternative, relative information content of the criteria, relative functional importance of the criteria. Based on experimental findings, [17] argue that human judgments of importance are related to impact rather than weighting.

Another strand of research acknowledges difficulties in weight assessment by focusing on easier questions for the DM, by including redundancy in elicited information, or by minimizing errors in weight determination. For instance, one frequently used approach is to let the DM compare different criteria in a pairwise fashion (as in the well-known Analytical Hierarchy Approach (AHP)) and to automatically determine suitable weights from this information (e.g., by the eigenvector method in the AHP, cf. [18]). Some other approaches use, for instance, an optimization approach (e.g., a linear programming formulation) for determining weights from the input of a DM taking into account possible inconsistencies. Two example studies from the last 50 years are [19] and [20].

A completely different idea, however, is not to use specific input from DMs for weight determination at all or the assessment of an overall objective function. Instead, knowledge from the outcome of previous decision situations could be used, for instance, in an approach based on machine learning. There are different scenarios of utilizing this idea: (1) If we assume that DMs (such as experts or experienced specialists for the considered problem situations) are able to make good decisions, data from previous problems and their selected decision alternatives could be used as training data for a suitable machine learning approach. (2) Another possibility would be that a DM evaluates a small number of alternatives in the context of a decision problem with a large number of alternatives. This input could then be used for the training task and subsequently the remaining alternatives are evaluated automatically. (3) It might also be possible not to use specific decisions as input to the training process but ex-post data from the decision situations. For instance, in an investment decision, knowledge about the benefit of different decisions (such returns from specific portfolios or assets) should become available at a later time.

The basic idea of using (supervised) machine learning in multicriteria decision making dates back to the late 1980s and was mostly related to the idea of employing neural networks in that context as discussed in a survey [21]. A recent example of using deep neural networks for providing results according to various established multicriteria approaches is analyzed in [22]. Of course, besides neural networks other standard machine learning approaches could be used as well. In their survey, [23], [24] discuss, for instance, the application of rule-based models and kernel-based models such as support vector machines. A systematic review of the combined usage of multicriteria analysis and machine learning with focus on deep learning and big data has been provided in [25] but without going deeper into the methodological issues of the considered studies. It might also be possible to use the machine learning approach in the context of parameter learning for established multicriteria methods (i.e., without any standard machine learning model), for combinations of such methods, or in combination with neural networks [26], [27]. This would better support the usage of prespecified multicriteria models or could improve transparency and explainability of the decisions.

Research related to the machine learning of preferences, in particular aiming at preference orders (expressed as binary relations such as partial or total orders) or rankings is also denoted as preference learning [28] and toolboxes have been suggested for that purpose [29]. Three types of ranking problems in preference learning are distinguished in [28]: label ranking, instance ranking, and object ranking. For the first two approaches, training alternatives are assumed with given pairwise rankings regarding associated labels, whereas object ranking uses training data directly from pairwise comparisons of the alternatives. Here, it is usually aimed at assigning a score to each instance which allows to derive an overall ranking.

It is a usual requirement of supervised machine learning that a sufficiently large set of training data is available. As further assumptions we may mention that the training data has emerged under comparable conditions, for instance regarding the preferences of decision makers over time. Also involving several DMs may lead to less consistent results if they have different perceptions or preferences regarding good solutions. Although the machine learning approach is based on an underlying optimization problem which can minimize such inconsistencies in training information resulting from human preferences, uncertainties in the decision problem and inherent dynamics, this at least affects the quality of outcomes as determined by standard measures such as accuracy, loss function, or F1 score.

More specific issues might result from the considered decision-making problem. Mostly, the above scenario of machine learning is considered for multiattribute decision making problems which are characterized by an explicitly given set of alternatives characterized by several attributes, criteria, or features which could be the direct input of an approach such as a neural network. The situation is more complicated with multiobjective optimization problems involving a set of alternatives characterized by constraints. Here, for instance, during various optimization problems occurring over time the constraints may change which could have a strong effect on the selected most preferred alternative. This is particularly relevant for our considered scenario.

In their review on machine learning in multiobjective optimization [30] states that "compared with other classical algorithms […] for solving MOPs, ML is a new method in this domain, and the number of relevant studies that focus on ML for MOO is much smaller". Nevertheless, they still determine more than 300 optimization-related publications. However, our further investigation shows that mostly the multiobjective aspects in respective publications are related to the optimization problem underlying the machine learning task or apply to a related problem (e.g., multiobjective feature selection). We, however, are interested in a multiobjective optimization problem with an underlying machine learning task which is rarely investigated in the literature.

We found a few publications which discuss objective functions that are computationally costly to evaluate (e.g., because of an underlying simulation model), where a substitute model resulting from machine learning could be used. Another aspect is considered in [31] where a problem of concrete mixture optimization with two or three objectives is discussed. Here, the machine learning part of the approach is for determining a complex and not explicitly given objective function. A similar approach regarding an asphalt mixture problem is studied in [32]. In [33], the problem of weight determination is discussed for more complex multiobjective problems as in systems design. Due to the complexities of the problem, e.g., regarding information on objective functions, it is suggested to use machine learning. They consider constraints on the weights, but more restrictive constraints such as in typical real-life combinatorial problems (as ours) are not considered in the approach. Summarizing, we can state that the problem of weight assessment by machine learning in the context of multiobjective optimization is so far not well investigated in the literature.

III. PROBLEM DESCRIPTION

Railway incidents can vary in scope and severity. They range from delays caused by peak passenger traffic, icy switches, and crew health to landslides and network power outages. While the largest of these incidents correspond to emergency situations with respective contingency plans, smaller incidents (i.e., disruptions) are resolved on the spot with situational solutions.

Some straightforward solution approaches are common to dispatchers to resolve smaller disruptions, e.g., how they utilize the schedule, tracks and trains, crews, and alternative services (e.g., busses). Most railway companies have adopted computer-aided systems and are increasingly looking to decision-support systems based on their operational data.

From an optimization perspective, the goal is to return a solution to an incident with changes to the original plan as cost-efficiently as possible (usually that is a combination of time and number of countermeasures/re-planning necessary), and subject to meeting all safety and regulatory requirements. Due to the situational complexity, the scope of decision variables requires a weighted target function, e.g., prioritizing passenger waiting time or the number of assets used. Any static optimization function would lead to unsatisfying results.

As such, this optimization problem differs from standard optimization models, such as master railway scheduling, job shop scheduling, and other settings where the number of variables is limited, the goal of cost optimization is clear, and the optimization problem does not differ situationally. There are two basic approaches to achieve situational optimization: rule-based systems (i.e., when the crew is sick, replace crew from available crew in alphabetical order), which become impractical with an increasing size of the planning problems, and systems learning from operational data, dispatcher behavior, and situations.

In an attempt to foster research and guide practice for the latter, we propose to represent dispatcher priorities in solving an incident management problem as weights (a_i) for performance indicators (p_i) used in an overall objective function of the mixed integer problem (1) to suggest solutions and to use the data gathered as inputs for a linear optimization that ensures regulatory compliance and optimality concerning the chosen weights.

Such an optimizer might use a minimalization similar to the following Eq. 1 where weighted (a_i) performance indicators (pi_i) calculated and minimized to find the best possible solution in regard to the weighted performance indicator.

$$min \left(\sum_i a_i pi_i\right) \tag{1}$$

TABLE 1. Assessed weights.

Weight	PI	Description
a_1	pi_1	The number of changed duties
a_2	pi_2	The number of staff journeys
a_3	pi_3	The number of vehicle changes
a_4	pi_4	The number of vehicle changes with little time
a_5	pi_5	The number of on-call staff changes
a_6	pi_6	The number of driver's cap changes with little time
a_7	pi_7	The relative occupancy rate of duties
a_9	pi_9	The number of changed service allocations

TABLE 2. Constant weights.

Weight	PI	Description
a_8	pi_8	The number of canceled services

where:

a_i : weight i

pi_i : performance indicator i

The weights a_i and the corresponding performance indicators p_i assessed in this project are listed in TABLE 1 and TABLE 2.

IV. EVOLUTIONARY ALGORITHMS

Since the 1960s, evolutionary Algorithms (EAs) have been introduced in several variants such as Evolutionary Programming, Evolution Strategies, and Genetic Algorithms for solving various complex optimization problems [34]. These algorithms mimic concepts from biological evolution and genetics, such as reproduction, mutation, recombination and crossover, and selection based on the fitness of individuals. EAs can be considered as one of the earliest contributions to what today is called nature-inspired algorithms and includes various other metaheuristics such as algorithms from the field of swarm intelligence.

EAs are based on a population of individuals (sometimes called chromosomes) which represent solutions to a given optimization problem or any problem-solving task in general. This set of solutions is usually initialized by random values in the beginning. Then, during a prespecified number of iterations (generations) or until some other termination condition is satisfied, the following iterative process is conducted: New solutions with random modifications are generated from the given population. For instance, two-parent solutions are used for creating two offspring solutions by using the information from both parents (with exchanging some information between parents). Besides this operation called crossover or recombination which exist in many variants, mutations occur that randomly change some of the parent information for the offspring solution. After that, only some of the solutions are considered for the next generation population, or before, only some solutions are selected for reproduction. To decide this, a fitness function is required which measures the quality of solutions. In an optimization problem, the fitness function is usually based on the given objective function, possibly with some modifications, e.g., for punishing violated constraints of the optimization problem. The selection then can be done in an elitist fashion where only the best solutions are selected or in a less exclusive way, e.g., by having a selection probability proportional to the fitness of a solution. In any way, the selection is expected to guide the otherwise random process toward better and better solutions.

As the general process conducted by EAs is very generic, it is possible to use these algorithms for a wide range of problems. In contrast to various traditional approaches in mathematical optimization, EAs do not require specific assumptions on the problem such as linearity, convexity or differentiability but can be used for a wide range of problems including hard-to-solve problem types, e.g., NP-hard combinatorial optimization problems as considered in our application. EAs can be considered as a robust black box optimization approach that does not require much knowledge of the considered problem. Nevertheless, it is often useful or even essential, to adapt and tune an EA in order to achieve good performance. For instance, the representation of solutions for the considered data structures should be wisely chosen, evolutionary operators such as recombination and mutation may require adaptation or parameters used in the EA should be tuned depending on a considered type of problem. These adaptations are usually made based on computational experiments, but there also mechanisms for self-adaptation were suggested [35].

For our present problem, the relationships of the variables to be determined (i.e., the weights of the multiple objective functions considered in the incident management problem) have a complex relationship with the utilized objective function. This objective function is based on an underlying combinational optimization problem which needs to be solved for each evaluation of our objective function. This complexity limits our knowledge of the problem and prevents using some standard optimization tools. An EA appears as a robust choice for exploring the problem and identifying suitable choices of the decision variables. They promise good opportunities to balance the costly search process between exploration of the search space and exploitation towards approximating a sufficiently good solution [36].

A. OUR EA IMPLEMENTATION

The following EA is inspired by reinforcement learning whereby the Optimizer and the incident cases are seen as the environment. The algorithm determines the weights of a fitness function for an ILP solver suggesting solutions to railway incidents. The ILP solver is used in a subordinate fashion to support the calculation of fitness values of the EA.

Fig. 1 depicts the general concept of the EA implementation. In A., multiple sets of solutions for the nine weights (decision variables) are randomly generated. B. assesses the fitness of each solution candidate for the entire training set of cases. Hereby, the optimizer with a weighted objective function is called for each case in the training set and the results are assessed by the fitness function of the EA. D. selects the best candidates of weight combinations and in

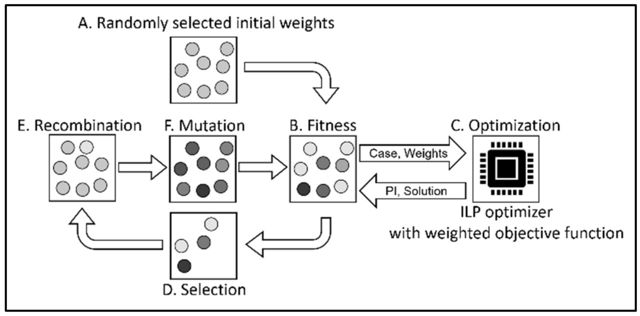

FIGURE 1. General concept of the EA inspired by reinforcement learning.

E., the best solutions are recombined. Finally, F. mutates all solution candidates with a decreasing degree over the generations before the new solutions are assessed again in B., against the training set with the optimizer.

B. EA FITNESS FUNCTION

The goal of the optimization process conducted by the EA is to determine weight values so that results are obtained corresponding to given reference results (e.g., results from historical data). Thus, the task can be considered as a supervised machine learning problem for the given specific model structure expressed by the underlying optimization problem (1).

The fitness function of the EA is based on the sum of the weighted squared differences between the optimized and the reference performance indicators over all cases. In order to have a fitness function to be maximized, a large deviation should correspond a small fitness value which is achieved by considering the reciprocal value of the sum of weighted squared differences.

$$\frac{1}{\sum_{j=1}^{n}\left(\sum_{i=1}^{9} C_i a_i \left(pi_{Opt_{ij}} - pi_{Ref_{ij}}\right)^2\right)} \tag{2}$$

where:

n : number of cases

C_i : helper variable if a_i is a variable or constant $C_i \in \{0, 1\}$

a_i : weight i

decision variable if $C_i = 1$

constant if $C_i = 0$

$pi_{Opt_{ij}}$: performance indicator i of case j resulting from the subordinate mixed integer problem (1)

$pi_{Ref_{ij}}$: reference indicator i of case j

PyGAD [37] is used to implement the EA algorithm. The open-source Python library is for building genetic algorithms and optimizing machine learning algorithms. PyGAD is fully customizable and allows the definition of customized fitness functions. However, PyGAD also supports pre-implemented crossover, mutation, and parent selection operators, which are fully customizable to the problem needs.

The fitness function is implemented according to Eq. 2. Furthermore, a customized mutation operation has been added which lowers the mutation grade with every generation. The formal description of the mutation operation is found in the following Eq. 3. The crossover type "uniform" is used, in which one parent is randomly selected for each gene out of the two mating parents to copy the gene. For the parent selection the steady-state selection is used. This selection type drops the worst solution candidatw1es of a generation and replaces them with new offspring solutions generated by crossover of the best solutions, while the rest of the population survives to the new generation.

$$mut\ (a_i) = a_i + x \tag{3}$$

where:

a_i : weight to mutate

x : follows a Gaussian distribution with a probability density $p\ (x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

σ : standard deviation for mutation $\sigma = \sigma_{init} * \frac{g_{max} - g_i}{g_{max}}$

The adaptive approach of calculating σ makes use of the following parameters:

σ_{init}: initial standard deviations, chosen as 0.05

g_{max} : maximal number of generations, chosen as 100

g_i : completed number of generations

The only further parameters for specifying the EA are the population size chosen as 10 and the mutation probability, which is chosen as 1 (i.e., each gene is mutated according to the mechanism described above). Moreover, some specifications for adapting the EA to the considered problem are used (i.e., the number of genes (9), the data types of the genes (float), and bound constraints for the genes (0.001 as lower bound and 1 as upper bound), in addition to specifications of the choices of genetic operators (selection, crossover) as discussed above.

V. RESULTS

All evaluation test sets were constructed based on five real-world test cases compiled by the industry partner. A larger number of test cases was not available due to significant manual effort in data extraction and preparation. However, the algorithm is evaluated in an ideal scenario, where all cases are solved with the same priorities for the performance indicators (same a_i for all cases for pi_i). This assumption represents an ideal situation, which is unrealistic in practical railway incident management. Therefore, additional synthetic test sets are created by randomly manipulating the performance indicators of the expected solutions in the base test set.

During each run of the EA, similar weights are determined, as depicted in Fig. 2. Furthermore, Fig. 2 compares the reference weights, which were used for original preparation of test cases (depicted in dark gray), with the trained weights (shown in light gray). The algorithm may not assess the same weights as the reference weights but with similar resulting performance indicators, as shown in TABLE 3.

However, the diversity of weight a_4 is the highest within these fifteen test runs. The weight a_4 is for the performance indicator pi_4 measuring the vehicle changes with little time.

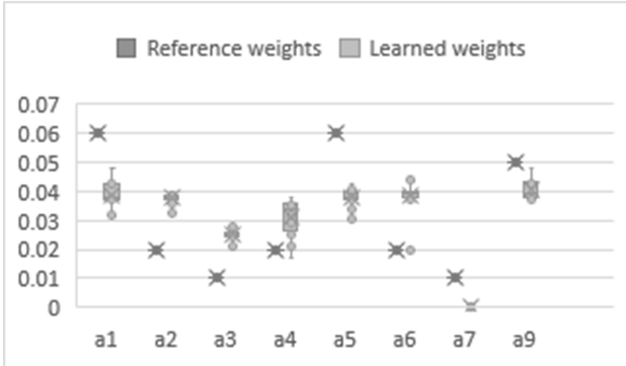

FIGURE 2. Learned weights of 15 runs on the base test set.

The test set offers only a small solution space in this regard and a_4 has therefore only little impact on the calculated solutions. Therefore, similar solutions are calculated with rather high and rather low weight in a_4.

TÂBLE 3 lists the test results of 15 training runs on the base test set. In 13 of 15 runs in total, the learned weights resulted in solutions generated by the optimizer with the expected performance indicators. However, in two of the 15 runs, the solution performance indicators are different for α_2 and α_9 in $case_5$.

The learned weights in run_4 and run_{11} result in solutions for $case_5$ with one lesser staff journey and one more changed service allocation. Therefore, the proposed algorithm finds weights that usually provide solutions with equal performance indicators as the training set and rarely finds weights where two performance indicators do not match the training set.

In contrast, the baseline weights deliver solutions where the performance indicators differentiate from the test set in three of the five cases for a_2, a_3, a_4 and a_9. Thus, the baseline solves fewer cases with fewer equal performance indicators than the proposed approach.

A. BASELINE

The baseline for the evaluation is built by distributing all learned weights equally. Therefore, the eight learned weights ($a_1, \ldots, a_7, a_9$) are set to 0.03125, and the constant weight of a_8 is set to 0.75.

B. BASE TEST SET

The base test set uses the same target weight distribution for each test case based on the assumption that every employee solves each case with the same company-wide priorities. The weight distribution is estimated by experts at the industry partners and represents ideal yet realistic priorities to solve cases. The algorithm is run 15 times on the base test set.

Fig. 3 illustrates the evolution of the weights' fitness. The graph depicts, as expected, a continuously improving fitness of the learned weights over 100 generations for all 15 runs. Therefore, the proposed approach outperforms the baseline in all 15 runs. The algorithm learns stable weights over

TABLE 3. Results of base test set.

	Cases with different performance indicators as expected by the training set							
run	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_9
1	–	–	–	–	–	–	–	–
2	–	–	–	–	–	–	–	–
3	–	–	–	–	–	–	–	–
4	–	$case_5$	–	–	–	–	–	$case_5$
5	–	–	–	–	–	–	–	–
6	–	–	–	–	–	–	–	–
7	–	–	–	–	–	–	–	–
8	–	–	–	–	–	–	–	–
9	–	–	–	–	–	–	–	–
10	–	–	–	–	–	–	–	–
11	–	$case_5$	–	–	–	–	–	$case_5$
12	–	–	–	–	–	–	–	–
13	–	–	–	–	–	–	–	–
14	–	–	–	–	–	–	–	–
15	–	–	–	–	–	–	–	–
B[a]	–	$case_1$ $case_4$	$case_4$ $case_5$	$case_1$ $case_5$	–	–	–	$case_5$

B[a]: Baseline

multiple runs where the optimizer calculates expected solutions regarding the performance indicators of the expected solutions.

C. SYNTHETIC TEST SETS

Due to the lack of available data, the base test set was randomly manipulated. These synthetic training sets better represent the real world, where not each case is solved with the same priorities for the performance indicators. Random manipulation may not replace actual test data but can be used to test the algorithm against varying priorities per case.

Different training sets are randomly created by randomly selecting one or two weights manipulated per case from the base test set. The manipulation of these weights can be interpreted as noise and is performed by adding random numbers determined from three different normal distributions and calculating the performance indicators of the solutions based on these weights. Eq. 3 shows the probability density for the Gaussian distribution.

$$mod\,(a_i, \sigma) = a_i + x \tag{4}$$

where:

a_i : weight to modify

x : follows a Ghaussian distribution with a probability density $p\,(x) = \frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

σ : standard deviation for modification For σ the following settings are considered: $\sigma_{min} = 0.01, \sigma_{med} = 0.03, \sigma_{max} = 0.1$

This method results in six synthetic training sets. First, two sets with small impacting changes $\sigma_{min} = 0.01$ on either one or two randomly affected performance indicators. Second, two sets with one or two medium significant ($\sigma_{med} = 0.03$) changes on one or two performance indicators. Finally, two sets with a maximum change ($\sigma_{max} = 0.1$) of one or two performance indicators.

Fig. 4 depicts the learning curve of weights over the generations. Again, the fitness of the runs continuously improves over the generations, as expected, and indicates convergence.

1) MINIMAL SYNTHETIC CHANGES ON THE TEST SET

In this test, one performance indicator at a time is manipulated to a small extent in each case (see TABLE 4). The genetic algorithm learns the weights to such an extent that only $case_1$ and $case_4$ provide a solution with diverging performance indicators. $Case_1$ is solved with one staff journey less and one more vehicle change with little time while $case_4$ is solved with two staff journeys less, two vehicle changes more, one additional change to on-call staff, and one change in service allocations less. Consequently, the algorithm learns sufficient weights in a minimal diverting training set.

In the next test, two performance indicators are manipulated to a minor degree in each case. The EA can still find reasonable weights where three cases are solved with the expected indicators. However, minor deviations are observed in $case_2$ and $case_5$. First, $case_2$ is solved with a much lower relative occupancy rate of duties. Second, $case_5$ is solved with one vehicle change less but a slightly lower occupancy rate. TABLE 5 shows the differences of the resulting performance indicators with the trained weights compared to the expected performance indicators in the synthetic training set.

Accordingly, even with two minor manipulations per case, convincingly learned weights can be expected to produce acceptable solutions.

2) MEDIUM SYNTHETIC CHANGES ON THE TEST SET

In the following test, only one random performance indicator of each case in the test set is randomly manipulated with a sigma equal to 0.03. Such a change in the performance indicators simulates diverse priorities for each case and it becomes more difficult for the algorithm to find fitting weights for all cases (see TABLE 6). However, most significant differences are observed by changing one priority in the performance indicator pi_9, the number of changed service allocations. The yield weights result in only solved $case_3$ with the aiming performance indicators and in all other cases significantly more changes of service allocations. The other performance indicators differentiate just slightly. For example, $case_1$ is solved with two more changed duties, three vehicle changes less and one vehicle change less with little time but 22 more service changes. $Case_2$ is solved with one more changed duty, two staff journeys less, and two vehicle changes less with little time but twelve more changed service allocations. Similar observations are in $case_4$ and $case_5$.

The algorithm managed to yield better results in the following test, where two priorities undergo a medium change of a sigma equal to 0.3 (cf. TABLE 7). $Case_3$ and $case_4$ are solved with the expected performance indicators and $case_5$ only with a lower relative occupancy rate. However, $case_1$ is solved with one staff journey less and one more vehicle change with little time, while all other performance indicators are as expected. $Case_2$ has a significant difference in service

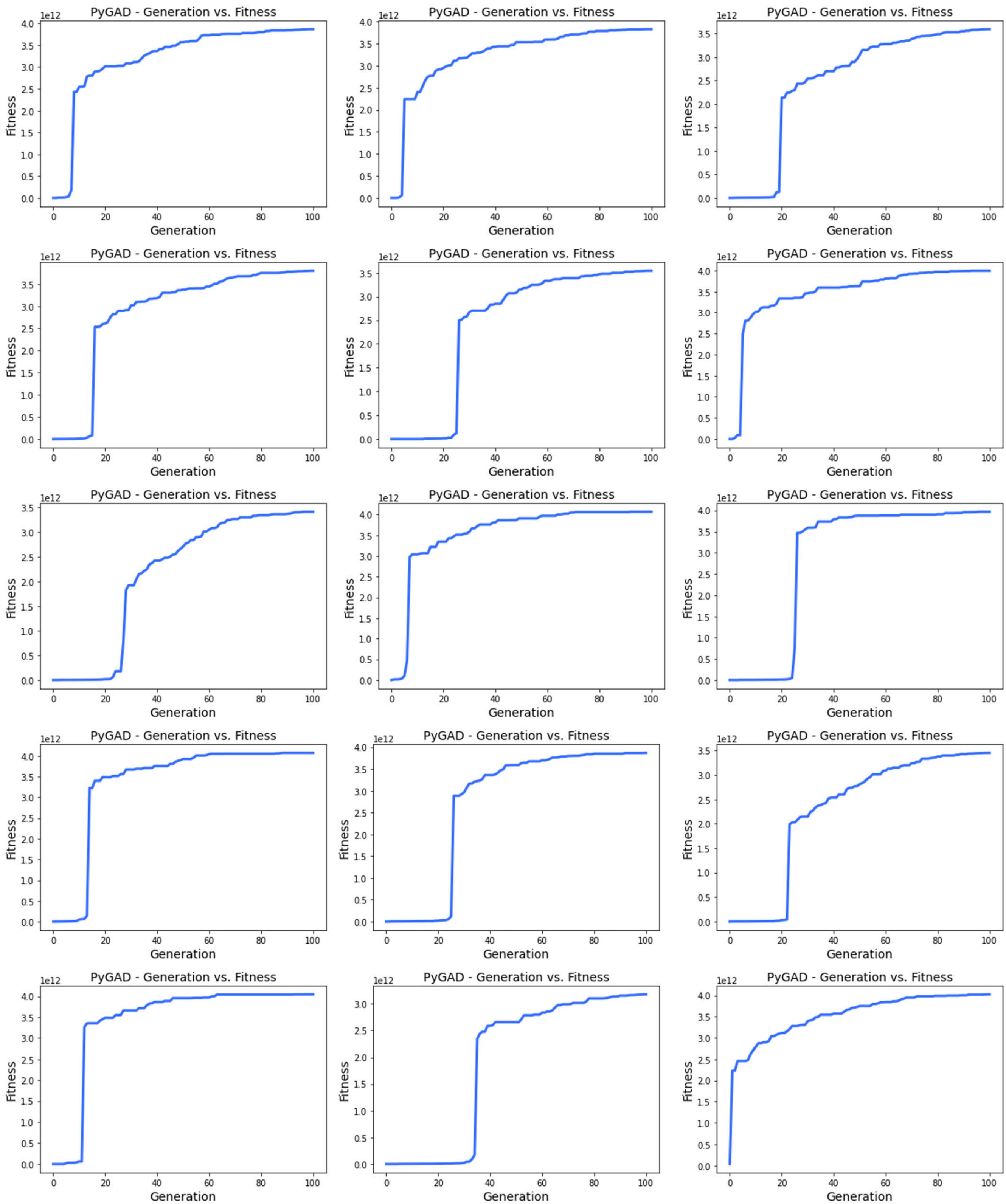

FIGURE 3. Fitness evolution of all 15 runs on base test set.

allocations of 24 fewer changes while the yield weights result in two fewer changes in duties, two more staff journeys, two more vehicle changes with little time and a slightly higher relative occupancy rate by 3%.

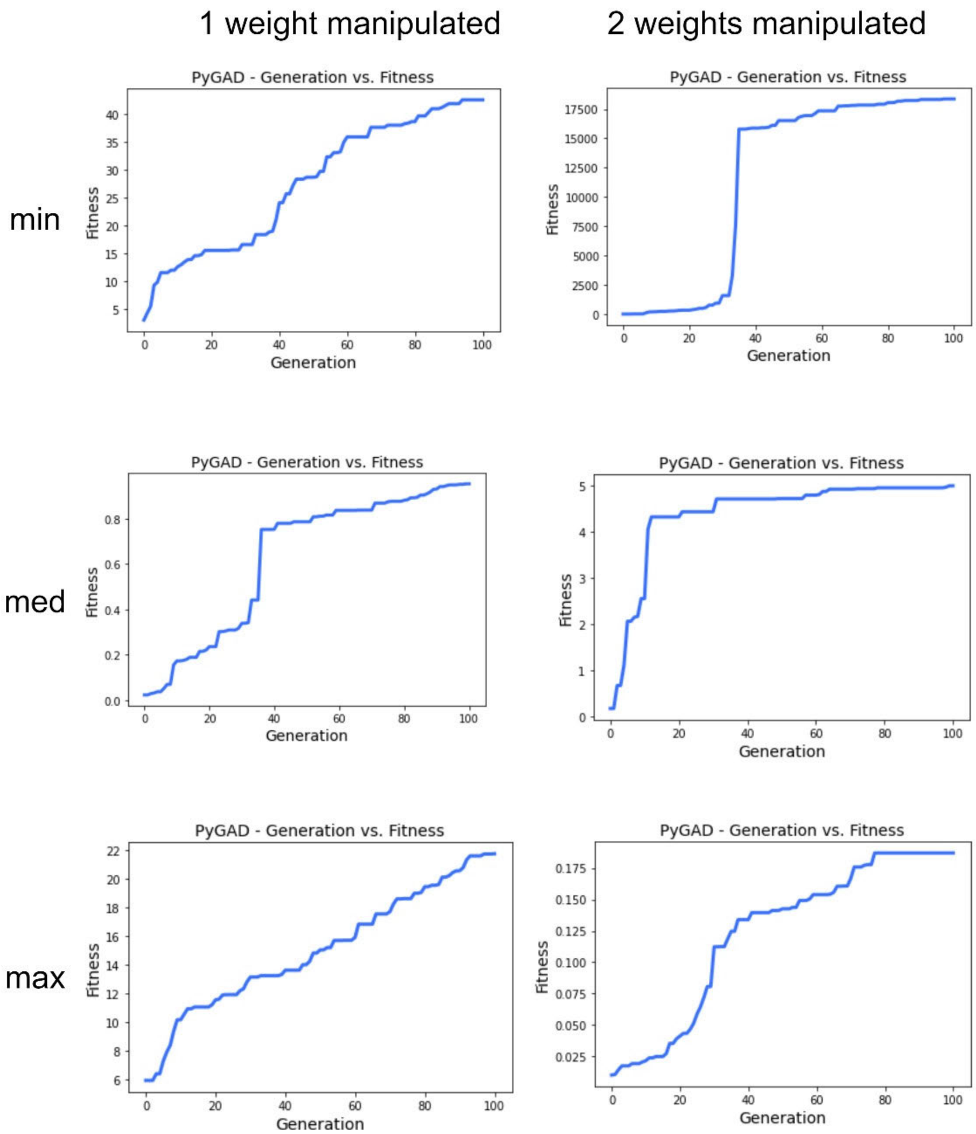

FIGURE 4. Fitness evolution on synthetic test sets.

The results of medium changes in one or two priorities illustrate the challenge in this domain. Moreover, as the priorities in individual cases differ, it becomes more difficult to find fitting weights for all cases. However, the algorithm can still find weights which serve all five cases in the small synthetic test set.

3) MAXIMUM SYNTHETIC CHANGES ON THE TEST SET

In this test, one performance indicator undergoes a strong manipulation of $\sigma_{max} = 0.1$ per case (see TABLE 8). These manipulations are more significant than the previous ones. The proposed approach still manages to learn reasonable weights where three of five cases are solved with the projected performance indicators. However, the performance indicators suffer in two cases, where the most significant difference is obtained in $case_5$ with three more staff journeys, two vehicle changes less, three vehicle changes with little time less and a slightly lower occupancy rate of duties by 4%. $Case_1$ on the other hand, is solved with only one staff journey less but one more vehicle change with little time.

Therefore, even under significant random changes and different priorities of the performance indicators per case, the algorithm still learns reasonable weights providing solutions with acceptable performance indicators.

TABLE 4. Results of minimal change on one PI.

	Differences								
case	pi_1	pi_2	pi_3	pi_4	pi_5	pi_6	pi_7	pi_8	pi_9
case 1	0	-1	0	1	0	0	0.00	0	0
case 2	0	0	0	0	0	0	0.00	0	0
case 3	0	0	0	0	0	0	0.00	0	0
case 4	0	-2	+2	0	+1	0	0.00	0	-1
case 5	0	0	0	0	0	0	0.00	0	0

TABLE 5. Results of minimal change on two PIS.

	Differences								
case	pi_1	pi_2	pi_3	pi_4	pi_5	pi_6	pi_7	pi_8	pi_9
case 1	0	0	0	0	0	0	0	0	0
case 2	0	0	0	0	0	0	-0.39	0	0
case 3	0	0	0	0	0	0	0	0	0
case 4	0	0	0	0	0	0	0	0	0
case 5	0	0	-1	0	0	0	-0.01	0	0

TABLE 6. Results of medium change on one PI.

	Differences								
case	pi_1	pi_2	pi_3	pi_4	pi_5	pi_6	pi_7	pi_8	pi_9
case 1	+2	0	-3	-1	0	0	0	0	+22
case 2	+1	-2	0	-2	0	0	-0.03	0	+12
case 3	0	0	0	0	0	0	0	0	0
case 4	-2	-3	+4	+1	0	0	-0.03	0	-28
case 5	2	-5	-6	-1	0	0	+0.05	0	+20

The final test set undergoes substantial changes, and the synthetic test set might be unreasonable (see TABLE 9). However, two performance indicators are manipulated randomly by a maximal modification of $\sigma_{max} = 0.1$ per case. The substantial modifications of two performance indicators are resulting in a challenging test set for the algorithm where the priorities between the cases drift far apart. The trained weights only solve one case without diversity to the training set. Even if $case_2$ and $case_3$ are not solved with an identical performance indicator, the differences to the training set are marginal. On the one hand, $case_2$ is solved with one staff journey more, two vehicle changes in total less where one is a vehicle change with little time, a higher staff occupancy of 3%, and one change less in service allocations. On the other hand, $case_3$ is solved with a lower staff occupancy of 68%, which is rather huge. In $case_1$ and $case_5$ the strongest deviations are observed. $Case_1$ is solved with 10 changed duties less, while all other performance indicators are as desired. Finally, in $case_5$ a huge difference to the expected performance indicators is observed. For example, the trained weights result in solutions of 39 staff journeys less and 50.5 changes in service allocations less. This huge difference is explained by the fact, that the expected solution cancels a full service, while the trained weights keep the train running, resulting in less changes than expected.

The results of this test support the finding that there is no one-size-fits-all solution when priorities drift far apart. This finding shows the importance of classifying cases in order to learn different weights for each case class. Such a case-based approach will only be possible if more detailed and standardized case data is available.

VI. DISCUSSION

Rail networks around the world are operating at maximum capacity, and are facing higher traffic due to population growth, and schemes to improve the attractiveness of rail travel over other modes of travel in line with ecological goals. With increasing traffic, the smallest disturbance in the network (railway incident) will spread faster than before and

TABLE 7. Results of medium change on two PIS.

	Differences								
case	pi_1	pi_2	pi_3	pi_4	pi_5	pi_6	pi_7	pi_8	pi_9
case 1	0	-1	0	1	0	0	0	0	0
case 2	-2	+2	0	+2	0	0	+0.03	0	-24
case 3	0	0	0	0	0	0	0	0	0
case 4	0	0	0	0	0	0	0	0	0
case 5	0	0	0	0	0	0	-0.41	0	0

TABLE 8. Results of maximum change on one PI.

	Differences								
case	pi_1	pi_2	pi_3	pi_4	pi_5	pi_6	pi_7	pi_8	pi_9
case 1	0	-1	0	+1	0	0	0	0	0
case 2	0	0	0	0	0	0	0	0	0
case 3	0	0	0	0	0	0	0	0	0
case 4	0	0	0	0	0	0	0	0	0
case 5	0	+3	-2	-3	0	0	-0.04	0	-2

TABLE 9. Results of maximum change on two PIS.

	Differences								
case	pi_1	pi_2	pi_3	pi_4	pi_5	pi_6	pi_7	pi_8	pi_9
case 1	-10	0	0	0	0	0	0	0	0
case 2	0	+1	-2	-1	0	0	+0.03	0	-1
case 3	0	0	0	0	0	0	-0.68	0	0
case 4	0	0	0	0	0	0	0	0	0
case 5	-2	-39	+12	0	0	0	+0.04	-1	-50.5

cause considerably higher economic damage through delays, missed connections and train cancellations.

Thus, there is a clear need to support dispatching teams with automated techniques, and the reduction of both the manual effort and the time duration to find a solution for handling an incident in railway operations are the most obvious requests of practitioners. In addition, the quality of a solution matters and is based on a number of objectives. Our discussion with railway companies led to 9 most relevant objectives which we considered in an aggregative model based on weighting these objectives.

The further discussion with practitioners has shown, that it appeared too difficult to specify these weights is a reliable way. In the routine dispatching praxis, there is too little time to consider the effects of different solutions for incident management on such objectives. Practitioners have little experience to effectively work with such objectives.

Therefore, we considered an approach of determining the weights based on historical data assuming that this is a plausible starting point when implementing an optimization approach for decision support. The results obtained with our approach appeared mostly plausible for practitioner and increased their understanding of the relevance of objectives and their dependence on the planning scenario.

It can be assumed that a permanent visualization of the objective values in their dispatching software (as planned by our software partner in the project) will further increase the consideration of objectives in the daily practice and provide human planners with a wider range of possibilities for exploring novel solutions in terms of the objective values which can be achieved. As the recording of incident cases with all relevant data including the resulting objective function values is integrated into the software used by several railway companies, we expect to have a much larger database in the future available for analysis.

We also expect that a future evaluation of planning data obtained from the enhanced approach will provide interesting insights into the priorities of planning objectives which will then be based on more transparency from the dispatching decisions. Last but not least, we expect future weights to deviate from those resulting from our study also due to the limitations of our current base test set.

VII. CONCLUSION

Railway operators' dispatching teams in charge of resolving railway incidents require decision support. We propose a novel approach to use an EA to train the weights of a weighted fitness function for the ILP solver that suggests solutions for railway incidents. The approach is tested on selected real-world railway incident management data.

From a practical point of view, this has two advantages: (a) the software developer/vendor does not have to configure the optimization problem to a myriad of possible scenarios, and (b) the solution weights can self-configure over time, leading to faster, and better decisions while using the same basic optimization algorithms and solvers.

Our approach outperforms a baseline where all weights are equally distributed. The proposed approach is most stable if the cases are solved with similar priorities, which lets us conclude that a single training set for all cases would be too broad to train the weights for a weighted ILP fitness function.

The current research is limited by two main factors: (1) treating the ILP as a black box, and (2) in-depth machine-based understanding of railway incidents. Such incidents should be enriched with contextual data, metadata and be formulated extendible as per the GA features. These limitations offer opportunity for further research.

First, more research is needed on optimization with self-configuring target functions, or more broadly on the intersection of classical optimization modeling and machine learning extensions of optimization components. This includes further analysis of the problem including performance measurement aspects. Also, the aspect of uncertainty in some of the data or dynamic changes of the situation (e.g., regarding the incident or replanning options) deserves further investigation and may suggest more refined solution procedures. Our findings also suggest that the specification of priorities and preferences in the planning process appears difficult and might suggest a more refined user support (e.g., by further empirical investigation, by training and consulting, or by a more elaborate user interface for the planning software). Second, railway operators must have proper incident management tracking practices and standardized data collection in place, embedded in their daily operations. From this, more-in depth and larger data sets could be used to refine and improve the proposed approach. Specifically, future research could focus on a case-based reasoning approach by organizing cases of railway incidents in different groups based on similarity measures and specializing the training sets for each case group, such as a new railway operator, or specific geographies, time to resolve, number of passengers affected, and various known seasonalities, e.g. time of the day ('rush hour'), day of the week, or seasons in the year.

ACKNOWLEDGMENT

The project was led by Qnamic AG, a leading provider of software solutions for resource planning and allocation for railway, supply, and other service companies. The practical inputs for railway incident management, and the test cases were provided by Société Nationale des Chemins de Fer Luxembourgeois (CFL), Luxemburg, and Schweizerische Südostbahn AG (SOB), Switzerland. The ILP was kindly contributed by the IMS Institute of Modeling and Simulation, University of Applied Sciences of Eastern Switzerland.

REFERENCES

[1] H. Alawad, S. Kaewunruen, and M. An, "Learning from accidents: Machine learning for safety at railway stations," *IEEE Access*, vol. 8, pp. 633–648, 2020, doi: 10.1109/ACCESS.2019.2962072.

[2] J. Szabo, S. Blandin, and C. Brett, "Data-driven simulation and optimization for incident response in urban railway networks," in *Proc. 16th Conf. Auton. Agents MultiAgent Syst.*, 2017, pp. 819–827. [Online]. Available: https://research.ibm.com/publications/data-driven-simulation-and-optimization-for-incident-response-in-urban-railway-networks

[3] D. An, Y. Ma, H. Song, W. Yang, Q. Li, and X. Du, "Research on optimization model and algorithm of railway wagon flow routing," in *Proc. Asia–Pacific Conf. Intell. Med. Int. Conf. Transp. Traffic Eng.*, Dec. 2018, pp. 45–49, doi: 10.1145/3321619.3321664.

[4] H. Bury, S. Spieckermann, D. Wortmann, and F. Hubler, "A case study on simulation of railway fleet maintenance," in *Proc. Winter Simul. Conf. (WSC)*, Dec. 2018, pp. 2851–2860, doi: 10.1109/WSC.2018.8632392.

[5] Y. Cui, Z. Tang, and H. Dai, "Case-based reasoning and rule-based reasoning for railway incidents prevention," in *Proc. Int. Conf. Services Syst. Services Manag.*, 2005, pp. 1057–1060, doi: 10.1109/icsssm.2005.1500156.

[6] H. Zhao, G.-H. Zhang, W. Dong, X. Yan, and X.-Y. Sun, "Fault diagnosis for rail turnout system based on improved introspective CBR method," in *Proc. 13th World Congr. Intell. Control Autom. (WCICA)*, Jul. 2018, pp. 989–994, doi: 10.1109/WCICA.2018.8630600.

[7] V. D. M. de Souza, A. P. Borges, D. M. V. Sato, B. C. Ávila, and E. E. Scalabrin, "Automatic knowledge learning using case-based reasoning: A case study approach to automatic train conduction," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 4579–4585, doi: 10.1109/IJCNN.2016.7727800.

[8] F. Petroski Such, V. Madhavan, E. Conti, J. Lehman, K. O. Stanley, and J. Clune, "Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning," 2017, *arXiv:1712.06567*.

[9] R. E. Rosenthal, "Concepts, theory, and techniques principles of multiobjective optimization," *Decis. Sci.*, vol. 16, no. 2, pp. 133–152, Apr. 1985, doi: 10.1111/j.1540-5915.1985.tb01479.x.

[10] B. F. Hobbs, "What can we learn from experiments in multiobjective decision analysis?" *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-16, no. 3, pp. 384–394, May 1986, doi: 10.1109/TSMC.1986.4308970.

[11] S. Schenkerman, "Use and abuse of weights in multiple objective decision support models," *Decis. Sci.*, vol. 22, no. 2, pp. 369–378, Mar. 1991, doi: 10.1111/j.1540-5915.1991.tb00352.x.

[12] K. Miettinen, F. Ruiz, and A. P. Wierzbicki, "Introduction to multiobjective optimization: Interactive approaches," in *Multiobjective Optimization* (Lecture Notes in Computer Science), vol. 5252. Berlin, Germany: Springer, 2008, pp. 27–57, doi: 10.1007/978-3-540-88908-3_2.

[13] B. Xin, L. Chen, J. Chen, H. Ishibuchi, K. Hirota, and B. Liu, "Interactive multiobjective optimization: A review of the state-of-the-art," *IEEE Access*, vol. 6, pp. 41256–41279, 2018, doi: 10.1109/access.2018.2856832.

[14] B. Afsar, K. Miettinen, and F. Ruiz, "Assessing the performance of interactive multiobjective optimization methods," *ACM Comput. Surv.*, vol. 54, no. 4, pp. 1–27, May 2021, doi: 10.1145/3448301.

[15] A. Morton and B. Fasolo, "Behavioural decision theory for multi-criteria decision analysis: A guided tour," *J. Oper. Res. Soc.*, vol. 60, no. 2, pp. 268–275, 2017, doi: 10.1057/PALGRAVE.JORS.2602550.

[16] E. U. Choo, B. Schoner, and W. C. Wedley, "Interpretation of criteria weights in multicriteria decision making," *Comput. Ind. Eng.*, vol. 37, no. 3, pp. 527–541, Nov. 1999, doi: 10.1016/S0360-8352(00)00019-X.

[17] T. Pajala, P. Korhonen, and J. Wallenius, "Judgments of importance revisited: What do they mean?" *J. Oper. Res. Soc.*, vol. 70, no. 7, pp. 1140–1148, Jul. 2019, doi: 10.1080/01605682.2018.1489346.

[18] T. K. Dijkstra, "On the extraction of weights from pairwise comparison matrices," *Central Eur. J. Oper. Res.*, vol. 21, no. 1, pp. 103–123, Jun. 2011, doi: 10.1007/s10100-011-0212-9.

[19] V. Srinivasan and A. D. Shocker, "Linear programming techniques for multidimensional analysis of preferences," *Psychometrika*, vol. 38, no. 3, pp. 337–369, Sep. 1973, doi: 10.1007/bf02291658.

[20] J. A. J. Mendes, E. A. Frej, A. T. D. Almeida, and J. A. D. Almeida, "Evaluation of flexible and interactive tradeoff method based on numerical simulation experiments," *Pesquisa Operacional*, vol. 40, pp. 1–25, Aug. 2020, doi: 10.1590/0101-7438.2020.040.00231191.

[21] T. Hanne, "Neural networks in multicriteria decision support," in *Multicriteria Decision Aid and Artificial Intelligence: Links, Theory and Application*. Chichester, U.K.: Wiley, Feb. 2013, pp. 101–126, doi: 10.1002/9781118522516.CH5.

[22] K. Martyn and M. Kadzinski, "Deep preference learning for multiple criteria decision analysis," *Eur. J. Oper. Res.*, vol. 305, no. 2, pp. 781–805, Mar. 2023, doi: 10.1016/j.ejor.2022.06.053.

[23] M. Doumpos and C. Zopounidis, "Preference disaggregation for multicriteria decision aiding: An overview and perspectives," in *New Perspectives in Multiple Criteria Decision Making*. Cham, Switzerland: Springer, 2019, pp. 115–130, doi: 10.1007/978-3-030-11482-4_4.

[24] M. Doumpos and C. Zopounidis, "Preference disaggregation and statistical learning for multicriteria decision support: A review," *Eur. J. Oper. Res.*, vol. 209, no. 3, pp. 203–214, Mar. 2011, doi: 10.1016/j.ejor.2010.05.029.

[25] M. Yang, S. Nazir, Q. Xu, and S. Ali, "Deep learning algorithms and multicriteria decision-making used in big data: A systematic literature review," *Complexity*, vol. 2020, pp. 1–18, Aug. 2020, doi: 10.1155/2020/2836064.

[26] T. Hanne, *Intelligent Strategies for Meta Multiple Criteria Decision Making*, vol. 33. New York, NY, USA: Springer, 2012, doi: 10.1007/978-1-4615-1595-1.

[27] O. Csiszár, G. Csiszár, and J. Dombi, "Interpretable neural networks based on continuous-valued logic and multicriteria decision operators," *Knowl.-Based Syst.*, vol. 199, Jul. 2020, Art. no. 105972, doi: 10.1016/j.knosys.2020.105972.

[28] J. Fürnkranz and E. Hüllermeier, "Preference learning and ranking by pairwise comparison," in *Preference Learning*. Berlin, Germany: Springer, 2021, pp. 1–17, doi: 10.1007/978-3-642-14125-6_4.

[29] V. E. Farrugia, H. P. Martínez, and G. N. Yannakakis, "The preference learning toolbox," 2015, *arXiv:1506.01709*.

[30] Q. Qu, Z. Ma, A. Clausen, and B. N. Jørgensen, "A comprehensive review of machine learning in multi-objective optimization," in *Proc. IEEE 4th Int. Conf. Big Data Artif. Intell. (BDAI)*, Jul. 2021, pp. 7–14, doi: 10.1109/BDAI52447.2021.9515233.

[31] J. Zhang, Y. Huang, Y. Wang, and G. Ma, "Multi-objective optimization of concrete mixture proportions using machine learning and metaheuristic algorithms," *Construction Building Mater.*, vol. 253, Aug. 2020, Art. no. 119208, doi: 10.1016/j.conbuildmat.2020.119208.

[32] C. Liang, X. Xu, H. Chen, W. Wang, K. Zheng, G. Tan, Z. Gu, and H. Zhang, "Machine learning approach to develop a novel multi-objective optimization method for pavement material proportion," *Appl. Sci.*, vol. 11, no. 2, p. 835, Jan. 2021, doi: 10.3390/app11020835.

[33] P. Wang, H. Zhu, M. Wilamowska-Korsak, Z. Bi, and L. Li, "Determination of weights for multiobjective decision making or machine learning," *IEEE Syst. J.*, vol. 8, no. 1, pp. 63–72, Mar. 2014, doi: 10.1109/JSYST.2013.2265663.

[34] T. Back, U. Hammel, and H.-P. Schwefel, "Evolutionary computation: Comments on the history and current state," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 3–17, Apr. 1997, doi: 10.1109/4235.585888.

[35] S. Meyer-Nieberg and H. G. Beyer, *Self-Adaptation in Evolutionary Algorithms* (Studies in Computational Intelligence), vol. 54. Berlin, Germany: Springer, 2007, pp. 47–75, doi: 10.1007/978-3-540-69432-8_3.

[36] X.-S. Yang, S. Deb, T. Hanne, and X. He, "Attraction and diffusion in nature-inspired optimization algorithms," *Neural Comput. Appl.*, vol. 31, no. 7, pp. 1987–1994, May 2015, doi: 10.1007/s00521-015-1925-9.

[37] A. Fawzy Gad, "PyGAD: An intuitive genetic algorithm Python library," 2021, *arXiv:2106.06158*.

PHILLIP GACHNANG received the master's degree in business information systems from the FHNW University of Applied Science and Arts Northwestern Switzerland and the master's degree in computer science from the University of Camerino, Italy. He has been a Research Assistant with the FHNW University of Applied Science and Arts Northwestern Switzerland, since 2019. From his business background, his main areas of expertise are system engineering, software development, and programming. His current research interests include computational intelligence, metaheuristics, optimization, system engineering, and software development.

JOACHIM EHRENTHAL received the M.Sc. degree in business administration from Mannheim University and the Ph.D. degree from the University of St. Gallen. He is currently a Professor in business information systems with the FHNW University of Applied Sciences and Arts Northwestern Switzerland. From his business background, his main areas of expertise are supply chain IT, distribution/last mile, and foreign trade. Having published on retail distribution and co-creation of value in supply chains, his current research interest includes accelerating the implementation of new supply chain technologies.

RAINER TELESKO received the Ph.D. degree in business informatics from the University of Vienna. He is currently teaching business analysis and requirements engineering for the bachelor's students in business information technology and the master's student in business information systems at the School of Business, FHNW University of Applied Sciences and Arts Northwestern Switzerland.

THOMAS HANNE received the master's degree in economics and computer science and the Ph.D. degree in economics.

From 1999 to 2007, he was with the Fraunhofer Institute for Industrial Mathematics (ITWM) as a Senior Scientist. Since then, he has been a Professor in information systems with the FHNW University of Applied Sciences and Arts Northwestern Switzerland and the Head of the Competence Center Systems Engineering, since 2012. He is the author of more than 200 journal articles, conference papers, and other publications, and he is an editor of several journals and special issues. His current research interests include computational intelligence, evolutionary algorithms, metaheuristics, optimization, simulation, multicriteria decision analysis, natural language processing, machine learning, systems engineering, software development, logistics, and supply chain management.