



Elevator group control as a constrained multiobjective optimization problem



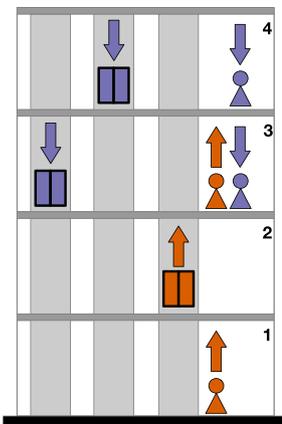
Aljoša Vodopija^{a,b,*}, Jörg Stork^c, Thomas Bartz-Beielstein^c, Bogdan Filipič^{a,b}

^a Department of Intelligent Systems, Jožef Stefan Institute, Ljubljana, Slovenia

^b Jožef Stefan International Postgraduate School, Ljubljana, Slovenia

^c Institute of Data Science, Engineering, and Analytics, TH Köln, Gummersbach, Germany

GRAPHICAL ABSTRACT



ARTICLE INFO

Article history:

Received 4 August 2020

Received in revised form 31 October 2021

Accepted 4 December 2021

Available online 10 December 2021

Keywords:

Elevator group control

S-Ring model

Multiobjective optimization

NSGA-II

Pareto front approximation

ABSTRACT

Modern elevator systems are controlled by the elevator group controllers that assign moving and stopping policies to the elevator cars. Designing an adequate elevator group control (EGC) policy is challenging for a number of reasons, one of them being conflicting optimization objectives. We address this task by formulating a corresponding constrained multiobjective optimization problem, and, in contrast to most studies in this domain, approach it using true multiobjective optimization methods capable of finding approximations for Pareto-optimal solutions. Specifically, we apply five multiobjective optimization algorithms with default constraint handling techniques and demonstrate their performance in optimizing EGC for nine elevator systems of various complexity. The experimental results confirm the scalability of the proposed methodology and suggest that NSGA-II equipped with the constrained-domination principle is the best performing algorithm on the test EGC systems. The proposed problem formulation and methodology allow for better understanding of the EGC design problem and provide insightful information to the stakeholders involved in deciding on elevator system configurations and control policies.

© 2021 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY license (<http://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Elevator systems gave architects new degrees of freedom and allowed buildings to become as multifaceted as they are today. The well-functioning of these transport systems is often taken

* Corresponding author at: Department of Intelligent Systems, Jožef Stefan Institute, Ljubljana, Slovenia.

E-mail address: aljosja.vodopija@ijs.si (A. Vodopija).

for granted in our modern, barrier-free life in urban areas. To realize this, most modern elevator systems are controlled by so-called elevator group controllers, which optimize the respective systems' service quality. Based on the passenger's desired destination, they assign moving and stopping policies to the elevator cars. Creating an adequate elevator group control (EGC) policy represents a complex problem, which involves multiple objectives and further depends on the elevator systems and building structure variables. The objectives include, besides passenger satisfaction, energy consumption and material attrition. The resulting multiobjective optimization function is highly nonlinear and multimodal, highly dynamic, and stochastic, mainly because passengers do not arrive in a deterministic manner, but based on a stochastic process. These problem properties render classic, gradient-based optimizers as not applicable and require advanced search strategies [1].

The presence of multiple conflicting objectives is a particularly notable characteristic of EGC. However, while EGC design is often referred to as a multiobjective optimization problem, it was approached with true multiobjective optimizers – in the sense of finding trade-offs between the objectives – only in our preliminary study in this domain [2]. In that study, we proposed a bi-objective problem formulation of the EGC optimization problem and used the sequential ring (S-Ring) model [3] to evaluate the solutions of the resulting optimization problem. Specifically, we dealt with two objectives that are often studied in the literature and both need to be minimized: the average number of floors with waiting passengers and the total number of elevator stops [4–6]. While the first objective is directly associated with passenger satisfaction, the second objective reflects both energy consumption and material attrition. It is worth noting that the two objectives are conflicting in nature since a prompt EGC service that would reduce the number of floors with waiting passengers requires many elevator car stops. A true multiobjective optimization approach is needed to obtain a set of trade-off solutions if preferences between the objectives are not known in advance. For this purpose, we applied multiobjective evolutionary algorithms (MOEAs) to find the fronts of trade-off solutions. The approach was tested on three real-world elevator systems where it confirmed the suitability of the methodology and offered relevant insights into problem properties.

However, a deeper analysis of the results obtained in [2] revealed that many produced solutions allow for a large number of elevator car skips. For example, in test problems reflecting residential buildings, passengers faced more than 20 elevator car skips, making such EGC policies inadmissible for practical use.

In this paper, we address the related issue by extending the initial bi-objective formulation with a constraint that limits the number of elevator car skips. The addition of this constraint fundamentally changes the EGC optimization problem and requires dedicated algorithms to solve the resulting constrained multiobjective optimization problem. In contrast to the related work, we do not combine the objectives into a single function through the weighted-sum approach, but use true constrained multiobjective optimization. In particular, we deploy five widely used multiobjective optimization algorithms (MOAs) featuring three constraint handling techniques (CHTs). The MOAs are systematically tuned to provide as unbiased results as possible. A new set of nine test elevator system configurations, ranging from simple to very complex ones, is composed to study the problem's hardness and test the applied algorithms' scalability. The results are thoroughly analyzed from the aspects of the solution quality and the algorithm performance, and specific problem characteristics. Finally, the superiority of using true multiobjective optimization over the weighted-sum approach is also demonstrated.

The paper is further organized as follows. Section 2 reviews the related work in EGC, focusing on various aspects of the

problem. Section 3 presents the S-Ring model used to simulate EGC and perform its optimization. Section 4 formulates EGC as a constrained multiobjective optimization problem. Section 5 describes the experimental setup used in the study, while Section 6 presents and analyzes the results. Section 7 concludes the paper by summarizing the work done, the essential findings and the key directions of future work in this domain.

2. Related work

In this section, we provide a detailed overview of the related work. We first address relevant aspects of dealing with EGC design including the issues of efficient control, changing environments, and passenger traffic forecasting. Next, special attention is devoted to the optimization of EGC systems. In particular, simulation-based optimization and optimization in the presence of multiple objectives are discussed. We conclude by presenting new concepts in the design of elevator systems.

2.1. Efficient control

Efficiency is a demanding issue in EGC. Computationally cheap methods become more and more important, especially in high-rise buildings with many passers-by. Banks of elevators are working in parallel in these buildings. They are required to serve the populace efficiently and quickly. For this purpose, Mahmud et al. [7] designed a control mechanism such that each call will be served by the elevator deemed to be the most energy-efficient. The control mechanism used was derived from the ground up using simple calculations to be computationally cheap, fast, and implementable with the most basic microcontroller. Similarly, Sahin et al. [5] presented a real-time monitoring system installed to reduce redundant stops and improve passenger comfort and energy consumption.

Because high-rise buildings require many elevators, they can be optimized group-wise. Utgoff and Connell [8] proposed an algorithm for dispatching cars to maximize efficiency for all the individuals who use an elevator in the group of elevators. Much information about the individuals in the group is inferred or estimated, greatly aiding the decision-making process. Insights are offered into the nature of various objective functions and their effects on system performance.

2.2. Changing environments

EGC strategies considering demand changes over time, different traffic patterns and differences between floors (or zones) are subject to on-going research. Fujino et al. [9] presented a concept for EGC systems that can change control settings according to individual floor utilization situations. The floor-attribute-based control method uses a combination of floor-attribute-based evaluation and car-attribute-based evaluation. The authors propose an online parameter tuning method using genetic algorithms. Pepyne and Cassandras [10] developed optimal dispatching controllers for elevator systems during peak traffic. A peak traffic period arises during the start of a business day in office buildings where many passengers are moving from the first floor up into the building. The cars deliver the passengers and then return empty to the first floor to pick up more passengers. The authors introduced a threshold-based policy that dispatches an available car from the first floor when the number of passengers inside the car reaches or exceeds a threshold that depends on several factors, including the passenger arrival rate, elevator performance capabilities, and the number of elevators available at the first floor. Because most elevator systems have sensors to determine the car locations and the number of passengers in each car, such demand-driven policies can be easily implemented in EGC systems. Dynamic programming techniques are standard tools for obtaining optimal control policies.

2.3. Passenger traffic forecasting

Robust dispatching decisions require that future passenger traffic is forecast based on the realized passenger flow in a building. EGC optimization can be combined with prediction methods. Thus, EGC dispatches elevators to passengers' calls in a dynamic environment where new calls continuously emerge. When making a dispatching decision, it is not known when and at which floors new passengers will register new calls, what is the number of passengers waiting behind these and existing calls, and what are their destinations. The problem is that this flow cannot be directly measured. However, it can be estimated by finding the passenger counts for the origins and destinations of every elevator trip occurring in a building. An elevator trip consists of successive stops in one direction of travel with passengers inside the elevator. Kuusinen et al. [11] formulated the elevator trip origin-destination matrix estimation problem as a minimum cost network flow problem and applied a branch-and-bound algorithm to solve it.

2.4. Simulation-based optimization

Optimization of real elevator systems is an important technique. In addition to elevator test stands (elevator towers), simulators are valuable tools in EGC optimization. For example, the multi-car elevator (MCE) simulator is a popular open-source implementation [12].

However, it is also of great interest to use a simulator which implements the essential features of an EGC system only and does not require the specification of too many details. Results from this essential simulator can be easily compared between various elevator implementations. This idea inspired Markon et al. [13,14] to develop the S-Ring (sequential ring). The S-Ring is a simplified model of a complex discrete dynamic system, i.e., an EGC system. The proposed model, which is a simplified model of elevator group control, has most of the properties that make the elevator group control problem challenging and popular, but in contrast to elevator models, it is simple and easily reproducible. It can be used as a benchmark for EGC optimization studies. The optimal control problem for the S-Ring can be formulated as a dynamic programming problem [15]. The S-Ring system is constructed in a way that balances between two conflicting requirements: it retains the most critical dynamic characteristics of the EGC system, but at the same time, it allows for exact solutions by algorithmic methods. As a concrete example, Markon [3] derived the S-Ring model from a formal model of EGC, presented the solution process, and, using the exact solution, benchmarked some optimization methods. He also described a variant, the S-Lane model, and showed its solution as an example of extending the technique to related problems.

In [1], Bartz-Beielstein et al. used the S-Ring to benchmark single-objective heuristics. Using the S-Ring model, it is possible to retain a high level of complexity and optimize an EGC control strategy using modern heuristics with a high number of strategy evaluations while keeping a feasible computational load.

Onat et al. [16] used the S-Ring for comparing different reinforcement learning schemes and stochastic approximation and Q-learning. Bartz-Beielstein et al. [14] demonstrated that a Nelder-Mead simplex algorithm and a quasi-Newton method could not escape from local optima while optimizing the S-Ring. In contrast to the artificial test functions from commonly used test-suites, real-world optimization problems often have many local optima on flat plateaus. The distribution of local optima in the search space of the S-Ring is unstructured. Therefore, these algorithms were unable to escape plateaus of equal function values. The analysis of S-Ring optimization reveals that evolution strategies are flexible optimization tools.

2.5. Multiobjective optimization

Ruokokoski et al. [17] studied the EGC problem arising in destination control, because in many standard EGC optimization methods, a routing aspect is not considered: decision variables specify only request-to-elevator assignments. The average waiting time and average journey time are used as objective functions in the comparisons. This example shows that the EGC optimization problem can be formulated in the context of multiobjective optimization.

Surprisingly, although the EGC optimization problems are widely discussed and known for involving conflicting objectives, they are seldom solved with multiobjective optimization. For example, Hakonen et al. [4] utilized a set of objectives, such as the passenger waiting time, the ride time, and the total number of elevator stops, but combined them linearly into a single objective. Tyni and Ylinen [6] used a weighted aggregation method to optimize the landing call waiting time and energy consumption with an evolutionary algorithm in a real-time environment.

In contrast, in [2], we for the first time addressed the EGC optimization problem with a true multiobjective optimization approach. In particular, we used the S-Ring model to evaluate the solutions for the bi-objective problem formulation and applied MOEAs to find the fronts of trade-off solutions.

2.6. New elevator concepts

Takahashi et al. [18] analyzed multi-car elevator systems, which are a revolutionary new technology for high-rise buildings [19–21]. In such a vertical transportation using a rope-less elevator, the design of linear motors with a high ratio of payload to self-weight becomes an impotent issue. The basic requirement for a linear motor for the rope-less elevator system is smooth motion, high driving force, and lightweight. Takahashi et al. investigated the optimal design of such motors for multi-car linear-motor elevator applications. Onat et al. [22] and Gurbuz et al. [23] performed a multiobjective optimization of these new motors using evolution strategies.

3. S-Ring model

The S-Ring is a discrete, nontrivial event system to simulate elevator group control [3]. It is highly adaptable and thus applicable to different use-cases while maintaining a low implementation effort. The system's primary focus is to model the operation of an elevator system by simulating the handling of passenger traffic and passenger serving. It is intended to serve as a dynamic, fast to evaluate, and computationally inexpensive system for optimization purposes, i.e., finding solutions for EGC systems that serve passengers in the fastest, most energy-efficient, and most comfortable way. Due to its low computational cost, the S-Ring can quickly evaluate a broad variety of EGC systems as benchmarks for the proposed multiobjective optimization approach. In general, the S-Ring consists of three key elements: state-space representation, state transition table, and control policy. These elements are briefly explained in the rest of this section to equip the reader with general information on S-Ring models. Additional details can be found in [3,24].

3.1. State-space representation

An S-Ring is represented by its nodes $i \in \{1, \dots, N_n\}$. Specifically, each floor contributes two nodes, i.e., one for the upward and one for the downward direction. Exceptions are the first and the top floors that each contributes a single node. Consequently, $N_n = 2n - 2$, where n is the number of floors in the EGC system.

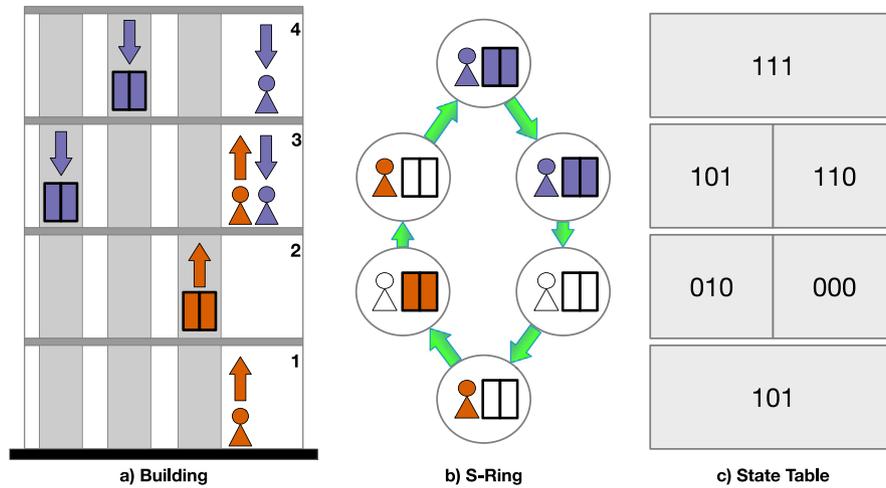


Fig. 1. S-Ring model with related building and state table. (a) shows a building with three elevator cars, four floors, and two waiting passengers for each direction. Upwards and downwards direction is possible for elevators and passengers and are colored red and blue, respectively. (b) illustrates the S-Ring model for this building, with a single node for the first and the top floors and two nodes for the rest. (c) displays the state encoding for each of the S-Ring nodes.

S-Ring nodes indicate the presence of passengers and elevator cars for a particular floor and direction. In more detail, the node state is determined by an ordered pair $s_i(t) = (c_i(t), e_i(t))$, where passenger state $c_i(t) \in \{0, 1\}$ indicates the presence of passengers and elevator state $e_i(t) \in \{0, 1\}$ the presence of an elevator car in node i at time $t \in \mathbb{N}$ (time is assumed to be discrete with unit interval steps). If $c_i(t) = 1$, we say the passenger state of node i is active. Similarly, $e_i(t) = 1$ indicates the elevator state of node i is active. The state of the whole S-Ring at time t is represented by a tuple $s(t) = (s_1(t), \dots, s_{N_n}(t))$ containing information on all node states. An example of an EGC system with the related S-Ring model is illustrated in Fig. 1. During the simulation, the maximum number of nodes with the active elevator state is equal to the number of available elevator cars, m . Moreover, the number of nodes with the active passenger state is dynamically changing over time and is influenced by the probability of a newly arriving passenger, p , and it is assumed to be identical at all floors and directions.

3.2. State transition table

The dynamics of an EGC system is expressed by state transition steps, $s(t) \rightarrow s(t + 1)$. The fixed and dynamic rules for state transition steps are shown in the state transition table (Table 1), which is applicable for each node in the S-Ring model [24]. For illustration, the node state $s_i(t) = (c_i(t), e_i(t))$ is extended by the next node state $s_{i+1}(t) = (c_{i+1}(t), e_{i+1}(t))$, forming a triplet $c_i(t), e_i(t), e_{i+1}(t)$, e.g., 101 for a waiting passenger (1xx), no elevator car in the current node (x0x) and a car in the next node (xx1). Moreover, the state transitions are influenced by the probability of arrival (p) or no arrival ($1 - p$) of a new passenger. Some states do not require or allow a policy decision and lead to predetermined transitions. For example, if the next node is occupied by another elevator car, the car in the current node cannot move forward until the elevator car in the next node moves forward. In each node, it is first checked whether a new passenger arrived. Next, if the current node has an active elevator state, the controller determines whether the elevator car stops or continues to the next node. Finally, the indication of the passenger state is updated depending on whether or not the passengers were served.

State transition steps happen sequentially in all N_n nodes in a simulation procedure that scans the nodes in a counterclockwise direction. Note that only one node state is changed by performing

Table 1

State transition table for the S-Ring. The table shows the node states and decisions to be made in certain node states. The transition probabilities are influenced by the probability of arrival (p) or no arrival ($1 - p$) of a new passenger and the control policy. The states are encoded in the following way: 1** indicates a passenger present, *1* an elevator in the current node, and **1 an elevator in the next node. Policy $\pi = 0$ indicates the elevator car passes the current node and $\pi = 1$ means the elevator car serves the current node. Some states have fixed transitions and do not require a policy decision.

Node state $s_i(t)$	Transition probability	Policy π	Node state $s_i(t + 1)$
000	$1 - p$		000
	p		100
100	1		100
010	$1 - p$		001
	p	0 1	101 010
110	1	0 1	101 010
	001	$1 - p$	
p			101
101	1		101
011	1		011
111	1		011

a single state transition step. The S-Ring simulation is highly influenced by the number of the performed state transition steps, N_t . This number has to be large enough in order for the simulation to converge.

3.3. Control policy

The control policy π establishes the decision of either serving or passing a passenger for the required state transition step and is crucial for any optimization. In general, an optimized policy could be realized by a lookup table if n remains very small. However, for large n this becomes infeasible, and a perceptron neural network is usually utilized in the literature to perform the state-to-decision mapping [24]. The perceptron has a simple, direct structure without hidden layers. It includes $2N_n$ binary inputs where the first N_n inputs denote the passenger states and the remaining N_n inputs the elevator states. The inputs are weighted (therefore the length of the weight vector equals $2N_n$) and directly connected to a binary output indicating the control

policy decision. For a given setup of n , m , and p , the perceptron's size remains fixed, and the state-to-decision mapping is only influenced by the perceptron weight vector $\vec{w} \in [-1, 1]^{N_n}$.

4. Problem formulation

In this section, we propose a multiobjective formulation of the EGC optimization problem. Specifically, we deal with two EGC objectives that are often studied in the literature, and both need to be minimized: the average number of S-Ring nodes with waiting passengers, and the total number of elevator stops [4–6]. In contrast to previous publications, we do not combine the objectives into a single function but adopt the multiobjective perspective. Moreover, to make it possible to compare the performance of elevator systems of various configurations (determined by the number of floors n and the number of elevator cars m), we consider normalized objective function values.

The first objective (h_1) is the proportion of nodes with waiting passengers. It is expressed as the average number of nodes with waiting passengers during the S-Ring simulation, M_w , divided by the number of all nodes, N_n :

$$h_1 = \frac{M_w}{N_n}. \tag{1}$$

The second objective (h_2) is the proportion of elevator stops. It is equal to the total number of elevator stops observed during the entire S-Ring simulation, M_t , divided by the maximum possible number of elevator stops. The latter can be calculated as the number of elevator cars m multiplied by the number of EGC simulation cycles, which in turn corresponds to the number of state transition steps, N_t , divided by the number of nodes, N_n , therefore

$$h_2 = \frac{M_t}{mN_t/N_n}. \tag{2}$$

Intuitively, the passengers' discomfort with long waiting times and long riding times due to many elevator stops does not increase linearly with time, but more drastically. To model this effect, we have additionally modified the original objectives as

$$f_1 = h_1^\alpha \quad \text{and} \quad f_2 = h_2^\beta, \tag{3}$$

where $\alpha, \beta \in [1, 2]$ are the objective function coefficients. The choice of their values is subjective, but the idea is to reflect the elevator system characteristics and the passenger preferences.

In our previous work on EGC [2], we observed that many obtained solutions allow for a large number of elevator car skips. It sometimes happened that a passenger was skipped more than 20 times, which made an EGC policy impractical. In this work we therefore introduce a constraint that renders all solutions with a large number of elevator skips infeasible. The corresponding constraint is expressed as the maximum number of elevator skips, M_s , that has to be less than or equal to M , therefore:

$$c = M_s \leq M. \tag{4}$$

The resulting constrained multiobjective optimization problem (CMOP) can be mathematically formulated as

$$\begin{aligned} &\text{minimize} \quad f_m(\vec{w}), \quad m = 1, 2 \\ &\text{subject to} \quad c(\vec{w}) \leq M \end{aligned} \tag{5}$$

where $\vec{w} = (w_1, \dots, w_D)^T$ is a perceptron weight, $f_1, f_2 : [-1, 1]^D \rightarrow \mathbb{R}$ are the two objective functions, $c : [-1, 1]^D \rightarrow \mathbb{R}$ is the constraint function, and $[-1, 1]^D$ the decision space of dimension $D = 2N_n$. Additionally, $f_m(\vec{w})$ is an objective value and $\phi(\vec{w}) = \max(c(\vec{w}) - M, 0)$ constraint violation.

Table 2

Characteristics of the test elevator system configurations: number of floors n , number of elevator cars m , probability of a newly arriving passenger p , number of nodes in the S-Ring representation N_n .

Config.	n	m	p	N_n
C1	5	1	0.01	8
C2	5	2	0.05	8
C3	5	3	0.10	8
C4	20	4	0.01	38
C5	20	6	0.05	38
C6	20	8	0.10	38
C7	50	10	0.01	98
C8	50	15	0.05	98
C9	50	20	0.10	98

5. Experimental setup

This section describes the experimental setup established to perform numerical experiments in EGC optimization. It describes the test elevator system configurations used in the experiments, the tested algorithms together with their CHTs, and the algorithm parameter tuning carried out for the sake of fair comparisons. It also provides the implementation details including the availability of the code.

5.1. Test elevator system configurations

Table 2 summarizes the characteristics of the test elevator system configurations used to assess the performance of multiobjective optimization approaches in our experiments. Nine configurations were carefully selected to cover a wide range of elevator systems operating in various kinds of buildings. The number of floors determines the building size. It varies among 5, 20, and 50. The probability of newly arriving passengers takes values among 0.01, 0.05, and 0.1. The smallest probability reflects the elevator systems operating in environments with low passenger traffic, such as residential buildings. On the other hand, the probability of 0.1 is connected to high passenger traffic observed in commercial buildings or parking garages. The number of elevator cars was selected based on building size and passenger traffic. The idea was to reflect real and meaningful elevator systems. The parameters α, β were set to 1.5 and M to 4 for all test configurations.

5.2. Tested algorithms

Based on the constrained multiobjective formulation of the EGC optimization problem, the experimental evaluation aimed at finding sets of trade-off feasible solutions representing Pareto front approximations. For this purpose we used five MOAs equipped with their default CHTs: Multiple Trajectory Search (MTS) [25] as an example of trajectory class MOAs, Multiobjective Particle Swarm Optimization (MOPSO) [26] as a representative of particle swarm MOAs, and three MOEAs, namely Nondominated Sorting Genetic Algorithm II (NSGA-II) [27], Differential Evolution for Multiobjective Optimization (DEMO) [28], and Multiobjective Evolutionary Algorithm based on Decomposition (MOEA/D) [29]. In addition, random search (RS) was run as a baseline optimizer to verify that the MOA results are meaningful.

The CHT incorporated in MTS handles the constraint as an additional objective function with binary values. If the constraint is satisfied, the value of this additional objective function is 0, otherwise 1. The rest of MTS is used as in the original algorithm.

The CHT used in MOPSO, NSGA-II and DEMO was the constrained domination principle (CDP) [27]. This approach extends the dominance relation and is one of the most widely-used techniques for constrained multiobjective optimization. CDP strictly

favors feasible solutions over infeasible ones. It ranks feasible solutions based on Pareto dominance and infeasible solutions based on constraint violation values. The formal definition of CDP, as introduced in [27], is provided with the following rule.

A solution \vec{w} is said to constrained-dominate a solution \vec{v} , if any of the following conditions is true.

1. Solution \vec{w} is feasible and solution \vec{v} is not ($\phi(\vec{w}) = 0$ and $\phi(\vec{v}) > 0$).
2. Solutions \vec{w} and \vec{v} are both infeasible, but solution \vec{w} has a smaller overall constraint violation ($\phi(\vec{w}) < \phi(\vec{v})$).
3. Solutions \vec{w} and \vec{v} are feasible and solution \vec{w} dominates solution \vec{v} ($\phi(\vec{w}) = \phi(\vec{v}) = 0$ and $\vec{w} \leq \vec{v}$).

In MOPSO, CDP is used during the update of the archive, where dominance relation is replaced with CDP, while NSGA-II and DEMO utilize CDP in the replacement phase, i.e., survivor selection. The rest of MOPSO, NSGA-II and DEMO is kept unchanged.

Finally, a CHT based on the penalty function was considered in MOEA/D [30]. In MOEA/D, the aggregation function (Tchebycheff function in our study) is enhanced with a penalty term as follows:

$$f^p(\vec{w} | \gamma) = f^{te}(\vec{w} | \lambda, z^*) + \gamma \phi(\vec{w}). \quad (6)$$

Here, f^{te} is the Tchebycheff aggregation function, $\gamma > 0$ the penalty weight, and ϕ constraint violation. This enhanced aggregation function is used to compare solutions in the update phase of MOEA/D.

5.3. Algorithm parameter tuning

The applied MOAs involve various parameters that effect their operation. The choice of parameter values can profoundly impact the algorithm performance. Thus, in many real-world applications, it is mandatory to select an adequate set of algorithm parameter values to solve the given problem efficiently. Besides, the adequate choice of parameter values allows for a more sound and robust comparison of various algorithms. This is because an algorithm's chance to be inferior due to an inappropriate choice of parameter values is highly reduced.

In the literature, various approaches to tuning or controlling algorithm parameters have been proposed. However, one of the most reliable and efficient methods used for real-world problems remains sequential model-based parameter optimization. This approach falls in the group of parameter tuning methods and is one of the most frequently used approaches when dealing with computationally demanding evaluations. In our work, Sequential Parameter Optimization (SPO) [31] was used to tune the algorithm parameters.

The target function optimized by SPO was the MOA performance measured as the obtained cumulative hypervolume given the algorithm parameter settings. The decision variables varied based on the MOA under consideration. In the case of MTS, the following parameters were tuned: the initial sample size, $n_s \in \{10, \dots, 20\}$, the number of local search iterations, $n_l \in \{80, \dots, 120\}$, the number of local search test iterations, $n_t \in \{6, \dots, 10\}$, the number of foreground solutions, $n_f \in \{3, 4, 5\}$. Next, the parameters tuned for MOPSO were: the population size, $n_p \in \{48, \dots, 500\}$, the mutation probability, $p_m \in [0, 1]$, the inertia weight, $w \in [0, 1]$, and the acceleration coefficients $c_1, c_2 \in [0, 4]$. Finally, for MOEAs the tuned parameters were: the population size, $n_p \in \{48, \dots, 500\}$, the crossover probability, $p_c \in [0, 1]$, and the mutation probability, $p_m \in [0, 1]$. Specifically, the scaling factor, $F \in [0, 2]$, was tuned instead of the mutation probability in DEMO.

Table 3

Tuned algorithm parameter values for MTS and MOPSO aggregated over test elevator system configurations of the same size: total number of solution evaluations f_e , initial sample size n_s , number of local search iterations n_l , number of local search test iterations n_t , number of foreground solutions n_f , population size n_p , number of generations n_g , mutation probability p_m , inertia weight w , and acceleration coefficients c_1, c_2 .

Config.	f_e	MTS				MOPSO					
		n_s	n_l	n_t	n_f	n_p	n_g	p_m	w	c_1	c_2
C1, C2, C3	10,000	13	102	6	3	268	38	0.52	0.55	1.34	0.61
C4, C5, C6	40,000	20	81	7	5	172	233	0.63	0.46	3.23	0.37
C7, C8, C9	100,000	18	114	10	4	208	481	0.42	0.49	0.47	1.50

For each algorithm, the number of generations, n_g , was determined as the number of solution evaluations, f_e , divided by the population size:

$$n_g = \left\lceil \frac{f_e}{n_p} \right\rceil. \quad (7)$$

Therefore, it was not considered as a decision variable (algorithm parameter) by the tuning process. The number of allowed solution evaluations changed proportionately with the number of floors: $f_e = 2000n$. Note that $n_g n_e$ may be greater than f_e . If this happened, the run was prematurely stopped after f_e solution evaluations.

Additionally, we used the following SPO configuration: Gaussian process regression for building surrogate models, classic differential evolution as the optimizer, Latin hypercube sampling (LHS) to initiate the MOA parameter values, and 25 target function evaluations (MOA runs) without repetitions. The tuned algorithm parameter values aggregated over configurations of the same size are summarized in Tables 3 and 4.

Finally, the candidate solutions in the algorithm runs were evaluated by the S-Ring simulation with a predefined number of simulation cycles. This was 100,000 for all test elevator system configurations. As a consequence, the number of state transition steps was equal to $N_t = 100,000N_n$, which was sufficient for the convergence of all S-Ring models used in this study.

5.4. Implementation

All the algorithms and functionalities needed for this study were implemented in the R programming language [32]. The experiments with MOPSO, NSGA-II and MOEA/D were carried out using the R packages `mopsocd`, `mco` and `MOEADr`, respectively. On the other hand, MTS and DEMO were reimplemented in R from scratch and are available at [33]. Finally, SPO, as implemented in the SPOT R package, was used in the tuning phase.

An R package including the code used in this study and an R vignette explaining all the functionalities can be found in the GitHub repository [33]. The package allows for further experimentation with additional elevator configurations and can be freely used under the conditions determined by the GNU General Public License, version 3 [34].

6. Results

This section reports on the results of the numerical experiments. It first presents the effectiveness and efficiency of all the tested algorithms. Based on the findings, it then analyzes the MOEA results in more detail, and finally scrutinizes the results of NSGA-II as the best performing algorithm. The section concludes by characterizing the EGC optimization problem.

Table 4

Tuned algorithm parameter values for MOEAs aggregated over test elevator system configurations of the same size: total number of solution evaluations f_e , population size n_p , number of generations n_g , crossover probability p_c , mutation probability p_m , and scaling factor F .

Config.	f_e	NSGA-II				DEMO				MOEA/D			
		n_p	n_g	p_c	p_m	n_p	n_g	p_c	F	n_p	n_g	p_c	p_m
C1, C2, C3	10,000	120	84	0.78	0.46	328	31	0.29	0.22	204	50	0.77	0.50
C4, C5, C6	40,000	240	167	0.79	0.16	264	152	0.55	0.34	120	334	0.59	0.32
C7, C8, C9	100,000	180	556	0.91	0.17	432	232	0.58	0.42	408	246	0.70	0.11

Table 5

Average cumulative hypervolume values for all tested algorithms on the test elevator system configurations.

Config.	RS	MTS	MOPSO	NSGA-II	DEMO	MOEA/D
C1	1.19 ± 0.00	1.19 ± 0.00	1.19 ± 0.00	1.19 ± 0.00	1.19 ± 0.00	1.19 ± 0.00
C2	1.10 ± 0.00	1.10 ± 0.00	1.10 ± 0.00	1.10 ± 0.00	1.10 ± 0.00	1.10 ± 0.00
C3	1.03 ± 0.00	1.04 ± 0.01	1.05 ± 0.00	1.05 ± 0.00	1.05 ± 0.00	1.05 ± 0.00
C4	1.09 ± 0.01	1.14 ± 0.01	1.16 ± 0.01	1.16 ± 0.00	1.16 ± 0.00	1.15 ± 0.01
C5	0.75 ± 0.02	0.86 ± 0.02	0.91 ± 0.01	0.95 ± 0.00	0.92 ± 0.01	0.92 ± 0.02
C6	0.62 ± 0.02	0.71 ± 0.02	0.75 ± 0.02	0.79 ± 0.00	0.76 ± 0.01	0.77 ± 0.01
C7	0.79 ± 0.05	0.87 ± 0.05	0.97 ± 0.05	1.03 ± 0.01	1.02 ± 0.02	0.98 ± 0.04
C8	0.50 ± 0.00	0.57 ± 0.03	0.59 ± 0.03	0.69 ± 0.02	0.61 ± 0.02	0.63 ± 0.02
C9	0.47 ± 0.01	0.55 ± 0.02	0.57 ± 0.03	0.63 ± 0.01	0.59 ± 0.02	0.60 ± 0.02

6.1. Algorithm effectiveness and efficiency

The algorithms were assessed from the point of view of both effectiveness (quality of results) and efficiency (required execution time). To assess the algorithm effectiveness, every algorithm was run 31 times, each time with a new randomly initialized population of solutions and the tuned parameter values from Tables 3 and 4. The cumulative hypervolume of the Pareto front approximation and the cumulative inverted generational distance plus (IGD⁺) were used to measure the quality of results. Cumulative means that all the nondominated feasible solutions found in the entire run were used for hypervolume and IGD⁺ calculation. Given $f_1, f_2 \in [0, 1]$, the reference point for hypervolume calculation was set to $(1.1, 1.1)^T$. Additionally, all the nondominated feasible solutions found over all runs and algorithms were used as a reference front for calculating IGD⁺.

The means of cumulative hypervolume and IGD⁺ values averaged over 31 runs are shown in Tables 5 and 6, respectively. The results indicate that RS performs comparably to MOAs on C1 and C2 according to hypervolume, and it underperforms on other test configurations. According to IGD⁺, RS is always outperformed by MOAs.

As we can see, for all five MOAs the obtained hypervolume and IGD⁺ values are very similar on C1–C4. In contrast, MOEAs perform noticeably better than MOPSO and MTS on C5–C9 concerning both hypervolume and IGD⁺, and MOPSO outperforms MTS. Besides, NSGA-II outperforms DEMO and MOEA/D on C5–C9 concerning IGD⁺. Moreover, NSGA-II performs better than DEMO and MOEA/D on C5, C6, C8 and C9 with respect to hypervolume, while on C7 only MOEA/D is outperformed by NSGA-II. Finally, only negligible differences are observed between DEMO and MOEA/D performance on C5, C6, C8 and C9 with respect to hypervolume and IGD⁺.

The statistical analysis confirms these findings. According to Friedman test, we observe statistically significant differences in MOA performance concerning both hypervolume ($\chi^2(4) = 31.29, p < 0.01$) and IGD⁺ ($\chi^2(4) = 23.556, p < 0.01$). Post hoc analysis with Wilcoxon signed-rank test and Benjamini–Hochberg procedure to adjust p -values shows that the three MOEAs significantly outperform MOPSO and MTS, and that MOPSO is superior to MTS. In addition, NSGA-II performance is superior to those of DEMO and MOEA/D, while no statistically significant differences are observed between DEMO and MOEA/D performance. The adjusted p -values of pairwise comparisons are shown in Table 7.

Table 6

Average cumulative IGD⁺ values for all tested algorithms on the test elevator system configurations.

Config.	RS	MTS	MOPSO
C1	0.0011 ± 0.0004	0.0008 ± 0.0004	0.0006 ± 0.0003
C2	0.0055 ± 0.0011	0.0032 ± 0.0007	0.0028 ± 0.0005
C3	0.0120 ± 0.0014	0.0067 ± 0.0012	0.0047 ± 0.0011
C4	0.0482 ± 0.0074	0.0134 ± 0.0038	0.0068 ± 0.0033
C5	0.1297 ± 0.0108	0.0438 ± 0.0087	0.0203 ± 0.0048
C6	0.1356 ± 0.0145	0.0592 ± 0.0073	0.0268 ± 0.0074
C7	0.1778 ± 0.0411	0.0820 ± 0.0098	0.0511 ± 0.0102
C8	0.1442 ± 0.0027	0.0957 ± 0.0108	0.0783 ± 0.0149
C9	0.1190 ± 0.0068	0.0537 ± 0.0113	0.0398 ± 0.0115

Config.	NSGA-II	DEMO	MOEA/D
C1	0.0007 ± 0.0002	0.0007 ± 0.0003	0.0006 ± 0.0002
C2	0.0022 ± 0.0006	0.0023 ± 0.0006	0.0021 ± 0.0005
C3	0.0025 ± 0.0005	0.0031 ± 0.0008	0.0036 ± 0.0018
C4	0.0059 ± 0.0026	0.0074 ± 0.0004	0.0088 ± 0.0040
C5	0.0090 ± 0.0025	0.0183 ± 0.0049	0.0195 ± 0.0064
C6	0.0094 ± 0.0025	0.0254 ± 0.0038	0.0180 ± 0.0059
C7	0.0118 ± 0.0047	0.0211 ± 0.0163	0.0476 ± 0.0034
C8	0.0221 ± 0.0110	0.0569 ± 0.0130	0.0635 ± 0.0085
C9	0.0118 ± 0.0041	0.0271 ± 0.0072	0.0287 ± 0.0108

Table 7

Adjusted p -values resulting from post hoc analysis with Wilcoxon signed-rank test and Benjamini–Hochberg procedure.

Pair	Hypervolume	IGD ⁺
MTS vs. MOPSO	0.0114*	0.0078*
MTS vs. NSGA-II	0.0056*	0.0078*
MTS vs. DEMO	0.0056*	0.0078*
MTS vs. MOEA/D	0.0056*	0.0078*
MOPSO vs. NSGA-II	0.0056*	0.0130*
MOPSO vs. DEMO	0.0056*	0.0342*
MOPSO vs. MOEA/D	0.0304*	0.0434*
NSGA-II vs. DEMO	0.0056*	0.0078*
NSGA-II vs. MOEA/D	0.0056*	0.0279*
DEMO vs. MOEA/D	0.9102	0.2031

*Indicate statistically significant differences (at the significance level of 0.05) in algorithm performance.

The execution times are reported in Table 8. As we can see, NSGA-II and DEMO are the most efficient among the considered MOAs, and MTS is slightly faster than MOPSO and MOEA/D. Besides, RS is significantly faster than MOAs. However, the most computationally expensive task is the S-Ring simulation, where many perceptron evaluations are required. In general, solutions

Table 8

Average execution times in seconds for all tested algorithms on the test elevator system configurations. The experiments were run on a 3.50 GHz Intel(R) Xeon(R) E5-2637V4 CPU with 64 GB RAM.

Config.	RS	MTS	MOPSO	NSGA-II	DEMO	MOEA/D
C1	5 ± 2	25 ± 2	40 ± 6	17 ± 0	16 ± 1	21 ± 1
C2	5 ± 2	36 ± 3	50 ± 7	23 ± 1	21 ± 1	46 ± 2
C3	5 ± 2	42 ± 6	64 ± 9	31 ± 3	28 ± 2	42 ± 5
C4	34 ± 4	355 ± 31	476 ± 82	325 ± 11	318 ± 5	461 ± 46
C5	43 ± 14	504 ± 46	573 ± 37	420 ± 11	395 ± 8	672 ± 115
C6	57 ± 33	643 ± 81	815 ± 95	493 ± 19	432 ± 13	787 ± 113
C7	115 ± 12	2356 ± 117	2567 ± 331	1870 ± 62	1998 ± 38	3019 ± 339
C8	150 ± 29	2483 ± 258	3346 ± 558	2232 ± 90	2128 ± 38	3249 ± 364
C9	203 ± 20	2992 ± 272	4834 ± 750	2562 ± 72	2415 ± 67	3283 ± 812

found by RS require only a negligible number of perceptron evaluations, but at the same time they are of low quality. Nevertheless, the EGC optimization considered here is a design problem, therefore efficiency is not of key importance.

In the rest of this section, the results of the three MOEAs are analyzed in more detail, while MTS and MOPSO are no longer considered since they were significantly less effective and efficient than MOEAs.

6.2. Detailed analysis of MOEA results

The results of MOEAs can be analyzed by visualizing the obtained Pareto front approximations. Fig. 2 shows Pareto front approximations for the test configurations resulting from typical runs of MOEAs. In more detail, all the runs corresponding to a given test elevator configuration are sorted based on the obtained cumulative hypervolume, and the front obtained in the median run is shown in the figure. We can see that the Pareto front approximations produced by NSGA-II are better in convergence for C5, C6, and C8 than those obtained by other MOEAs. In contrast to the statistical results, it is hard to see any difference in MOEA performance by only viewing the fronts for test configurations C7 and C9.

Fig. 3 shows box plots for the cumulative hypervolume obtained before and after parameter tuning on test elevator configurations C7 and C9. The box plots on the left show the results for the ten runs used in the initial phase of SPOT. These parameter values were sampled using the LHS design of experiment method and denote the results before tuning. The box plots on the right side show the hypervolume achieved in 31 runs with the tuned parameter values from Tables 3 and 4.

The results show that NSGA-II is less sensitive to parameter tuning than DEMO and MOEA/D. While the performance of the latter two algorithms varies considerably, NSGA-II is robust over various parameter settings. In particular, the results obtained before tuning NSGA-II are already concentrated around the median hypervolume values. The improvements in the tuning process for NSGA-II are negligible compared to the improvements obtained for DEMO and MOEA/D. For example, on C7, the hypervolume values obtained before tuning NSGA-II are almost as good as those obtained after tuning it. Moreover, the results obtained before tuning NSGA-II are already better than the results obtained after tuning MOEA/D. We can also see that DEMO sometimes performs better than NSGA-II but at the expense of performing worse in most runs.

The box plots for C9 show that NSGA-II performs better than DEMO and MOEA/D even with random parameter settings. Additionally, it is worth noting that NSGA-II using tuned parameters produces three outlier runs: one run achieving much better results and two runs achieving much worse results than the median run. Test elevator configuration C9 is the hardest to solve, and it seems that even the best performing algorithm can sometimes underperform on it.

In Fig. 4, the means of cumulative hypervolume progress are shown for C7 and C9. The x -axis indicates the spent solution evaluations and y -axis the corresponding cumulative hypervolume values. Although NSGA-II and DEMO achieve comparable hypervolumes on C7, NSGA-II is more efficient. NSGA-II needs about 50,000 solution evaluations to converge, while DEMO needs the whole computational budget of 100,000 solution evaluations to obtain the same result. Similar behavior is observed for other test elevator configurations. This further proves that NSGA-II is the best performing algorithm on the test elevator system configurations used in this study.

To summarize, NSGA-II performs at least as well as DEMO and MOEA/D on C1–C4 and C7, and outperforms both algorithms on C5, C8 and C9. Its efficiency and robustness are also superior to other MOEAs (Figs. 3 and 4). For this reason, we further analyze the NSGA-II results.

6.3. Further analysis of NSGA-II results

We investigated how NSGA-II performs concerning both objectives. In other words, we analyzed the spread of the obtained Pareto front approximations. We created two constrained single-objective optimization problems, where the first (f_1) and the second (f_2) objective were subject to minimization, separately. Ordered pairs of solutions for both objectives represent approximations of ideal points. To obtain these approximations, the best feasible solutions found in 31 runs of classic differential evolution (DE) [35] were recorded.

The results are shown in Fig. 5, where both optimized solutions found by DE and the Pareto front approximations obtained in the median runs of NSGA-II are depicted. From the figure it is evident that it is harder for NSGA-II to find optimal solutions with respect to the second objective, e.g., for C7 and C8. This was expected since this objective is strongly negatively correlated with the constraint (see Fig. 7). On the other hand, we can see that NSGA-II can always find near-optimal solutions concerning the first objective. In conclusion, a typical run of NSGA-II can find a satisfactorily spread Pareto front approximation for each test elevator configuration.

To validate the proposed approach, we also performed an external comparison between NSGA-II and a state-of-the-art method for EGC optimization. In the related work, multiobjective EGC optimization problems are solved using the weighted-sum (WS) approach [4–6]. This method combines the two objectives into a single objective as follows:

$$f^{ws}(\vec{w} | \gamma) = \gamma f_1(\vec{w}) + (1 - \gamma) f_2(\vec{w}), \quad (8)$$

and transforms the bi-objective EGC optimization problem into a single-objective one:

$$\begin{aligned} &\text{minimize } f^{ws}(\vec{w} | \gamma) \\ &\text{subject to } c(\vec{w}) \leq M. \end{aligned} \quad (9)$$

Normally, WS requires a predefined γ that specifies the preference between the objectives. However, we wanted to find a

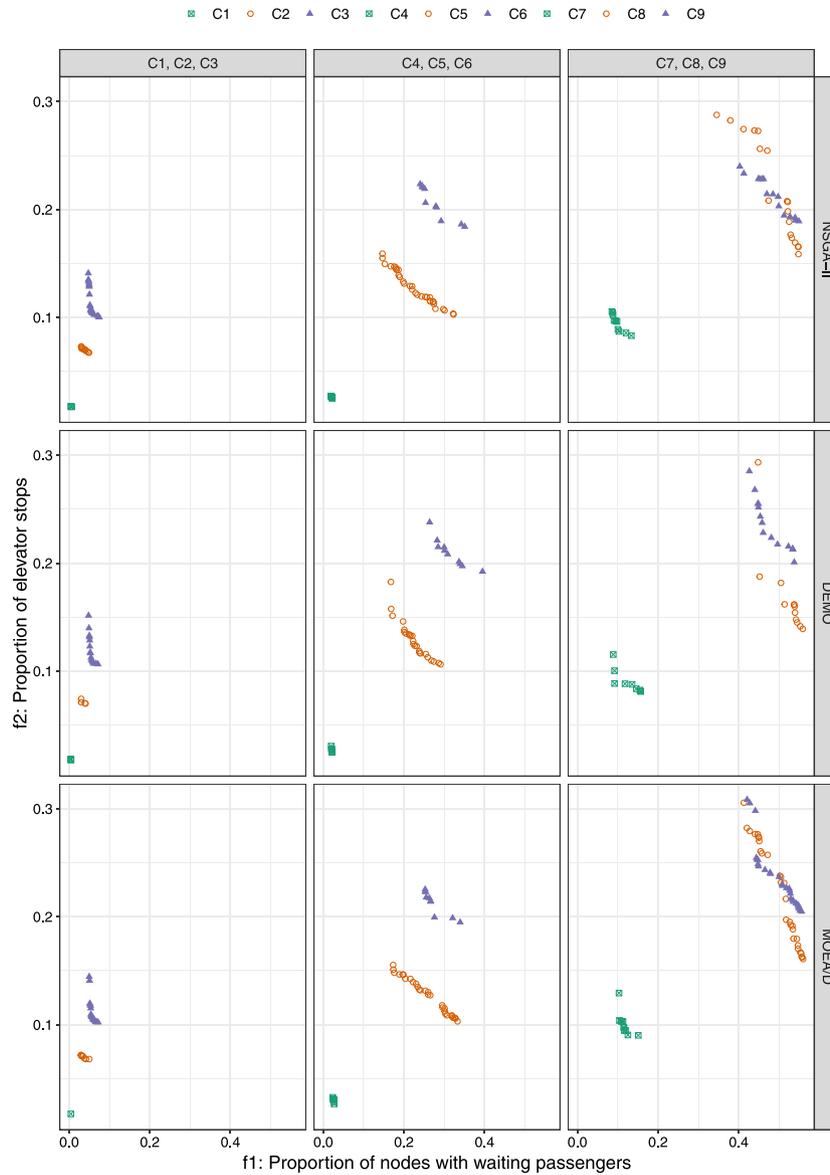


Fig. 2. Pareto front approximations for the test elevator system configurations. The first column shows the fronts for test configurations C1–C3, the second column for C4–C6, and the third column for C7–C9. Each row represents one algorithm, from top to bottom: NSGA-II, DEMO, and MOEA/D.

set of trade-off feasible solutions representing approximations for Pareto fronts. For this purpose, we used eleven values for $\gamma \in \{0, 0.1, \dots, 0.9, 1\}$ specifying various preferences between the objectives and resulting in eleven subproblems (9) for a given test configuration. For the sake of fair comparison, WS used the same amount of solution evaluations as MOAs (Tables 3 and 4). Specifically, the total amount of solution evaluations used for a test configuration was equally divided among eleven subproblems. For example, 10,000 solution evaluations were used to solve C1–C3 by MOAs. For this reason, only 909 solutions were allocated to each subproblem (9). We employed DE with a penalty function approach as a constraint handling mechanism to solve the resulting optimization subproblems. Like for MOEAs, the most influential parameters of DE (population size, number of generations, crossover probability and scaling factor) were also tuned using the procedure described in Section 5.3.

The results are summarized in Fig. 6. It shows the Pareto front approximations obtained in median runs of NSGA-II and WS. As expected, the Pareto front approximations obtained for C1–C3 are

comparable. On the other hand, the fronts obtained by NSGA-II on C6–C9 dominate those obtained by WS on these problems. Moreover, WS cannot always find 11 nondominated solutions (one per each γ value). These observations confirm the advantage of using true multiobjective optimization over the WS approach. Nevertheless, if the preference between the two objectives is known in advance, one could still use WS instead.

6.4. Problem characterization

We finally focused on the characterization of the EGC optimization problem with respect to the difficulty of handling individual elevator system configurations and correlations between the objectives and the constraint. We used the results found by RS to assess the complexity of the test configurations. As we have already seen, on C1 and C2, the hypervolume values obtained by RS are identical to those obtained by MOEAs (Table 5), indicating that these elevator configurations are trivial to

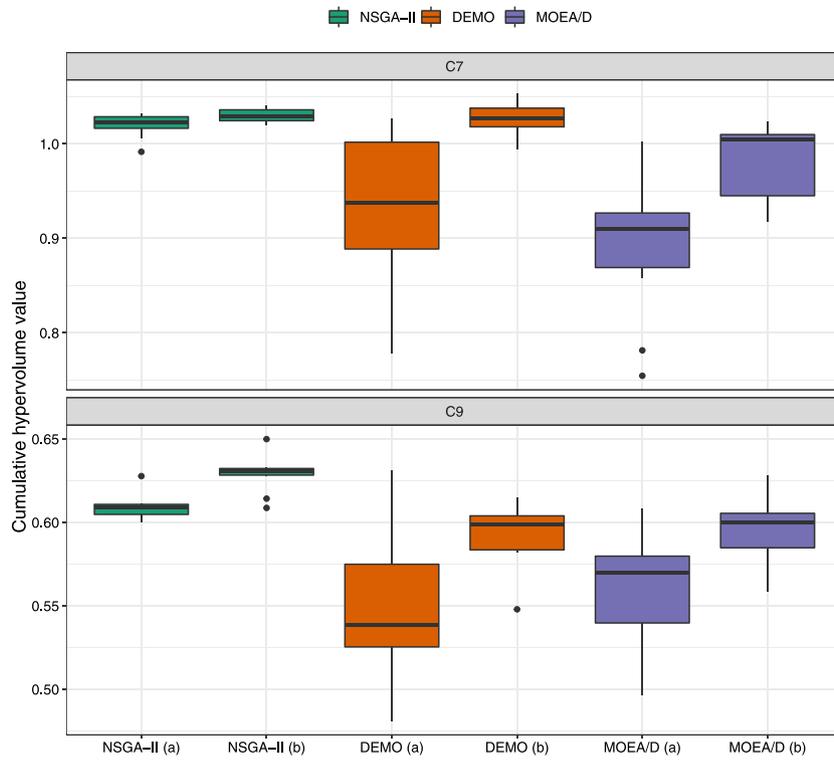


Fig. 3. Box plots of cumulative hypervolume values obtained by NSGA-II, DEMO and MOEA/D on test elevator system configurations C7 (top) and C9 (bottom). The cumulative hypervolume values obtained before (a) and after (b) tuning are shown for each algorithm.

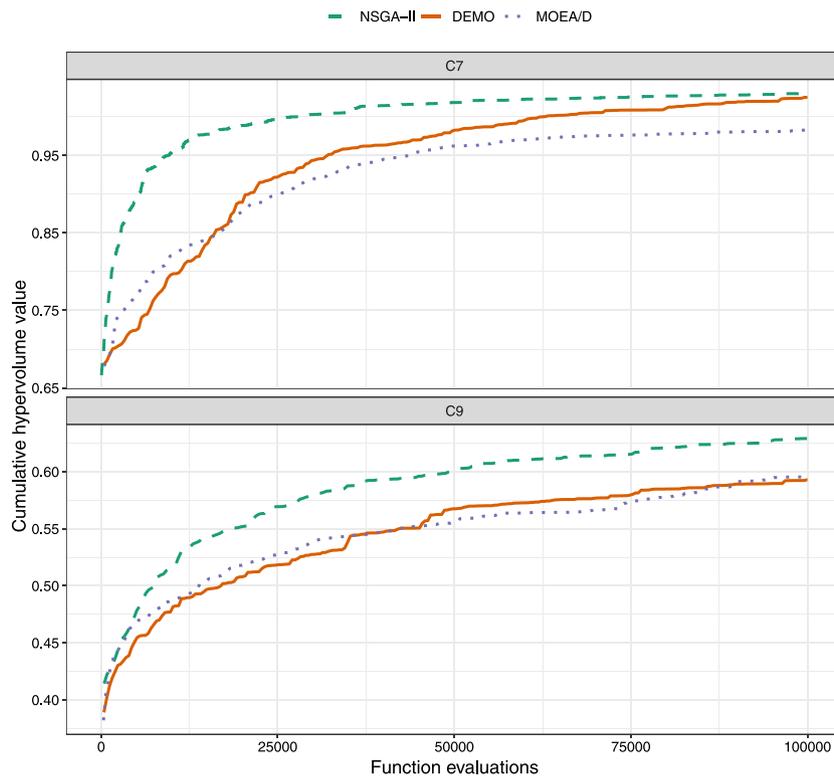


Fig. 4. Cumulative hypervolume progress for the test elevator system configurations C7 (top) and C9 (bottom).

solve. Indeed, C1 and C2 reflect the elevator systems operating in small buildings with low passenger traffic. Simple EGC policies,

e.g., never skip a passenger, are already suitable solutions quickly found even by RS.

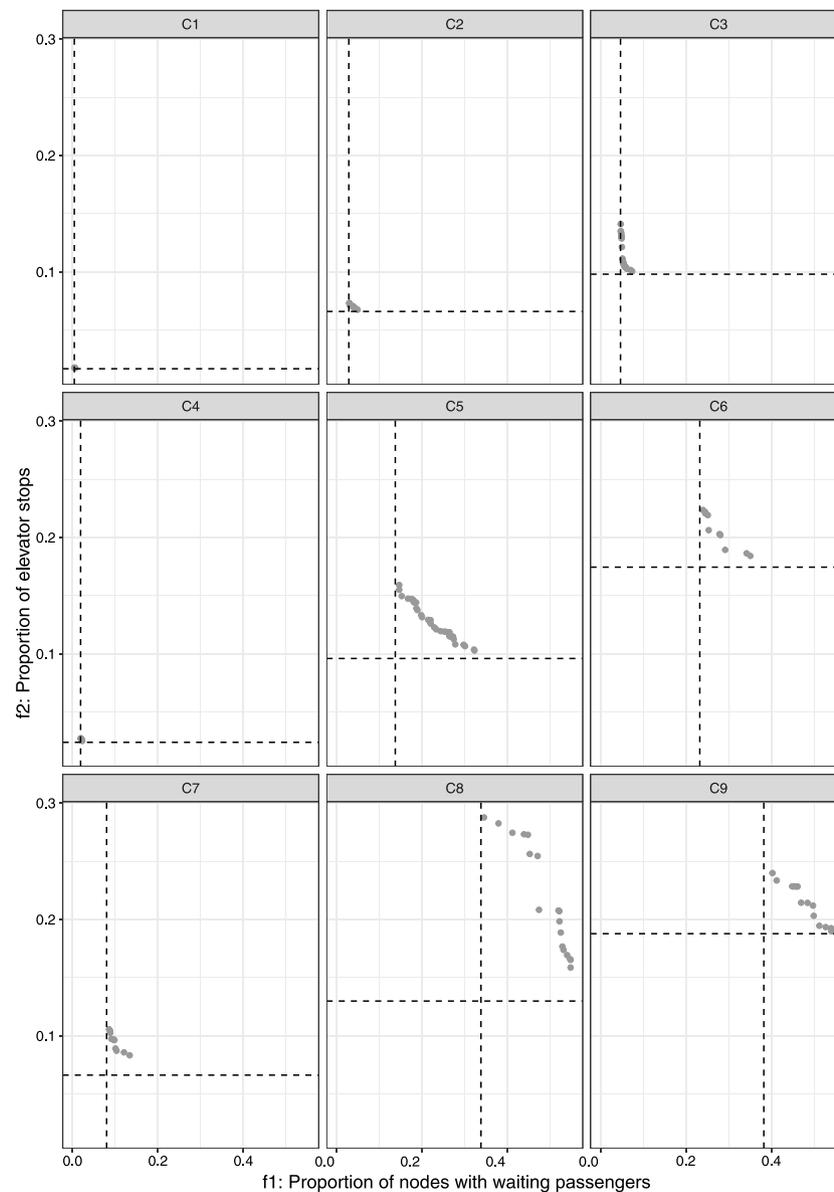


Fig. 5. Pareto front approximations resulting from typical runs of NSGA-II. The dashed lines show the minimum objective values found by DE, where each objective was optimized separately. The intersection of the vertical and horizontal lines represents an approximation for the ideal point. The first row shows the results for C1–C3, the second row for C4–C6, and the third row for C7–C9.

In contrast, on C3 and C4, the RS algorithm performs worse than MOEAs, obtaining about 2%–6% lower hypervolume (Table 5). The same observations are more intensely reflected on C5–C9, where RS obtains about 21%–27% lower hypervolume than MOEAs (Table 5). These results show that test configurations C5–C9 are harder to solve than C1–C4 and provide an insight into why we observe substantial differences in MOEA performance only on C5–C9.

To further investigate the difference between C1–C4 and C5–C9, we analyzed correlations between the objectives and the constraint. Fig. 7 shows these correlations for test configurations C2 and C5. For each configuration, 100,000 solutions were randomly sampled. As we can see, in the case of C2 the first objective (f_1) is strongly positively correlated (Pearson correlation coefficient $r = 0.89$) with the constraint (c). The EGC policy that always serves a passenger is a reasonable choice for an elevator system such as C2. This renders the test configuration C2 easy

to solve since optimizing the first objective already improves the constraint violation.

On the other hand, in C5 the correlation between the first objective and the constraint is weak (Pearson correlation coefficient $r = -0.31$). In more detail, we still observe a positive correlation between the objective and the constraint for $10 \leq c \leq 50$, but while this positive correlation persists until $c \leq 4$ in C2, this is not true in C5. This makes this constraint harder to satisfy than the constraint involved in C2.

As expected, the second objective (f_2) is in both cases strongly negatively correlated with the constraint (Pearson correlation coefficient $r = -0.71, -0.89$). The results for other test elevator configurations are not shown here since C1, C3, and C4 behave similarly to C2, and C6–C9 behave almost identically to C5.

7. Conclusions

We explored the optimization of EGC, which is a task relevant in the design and operation of multi-car elevator systems. The

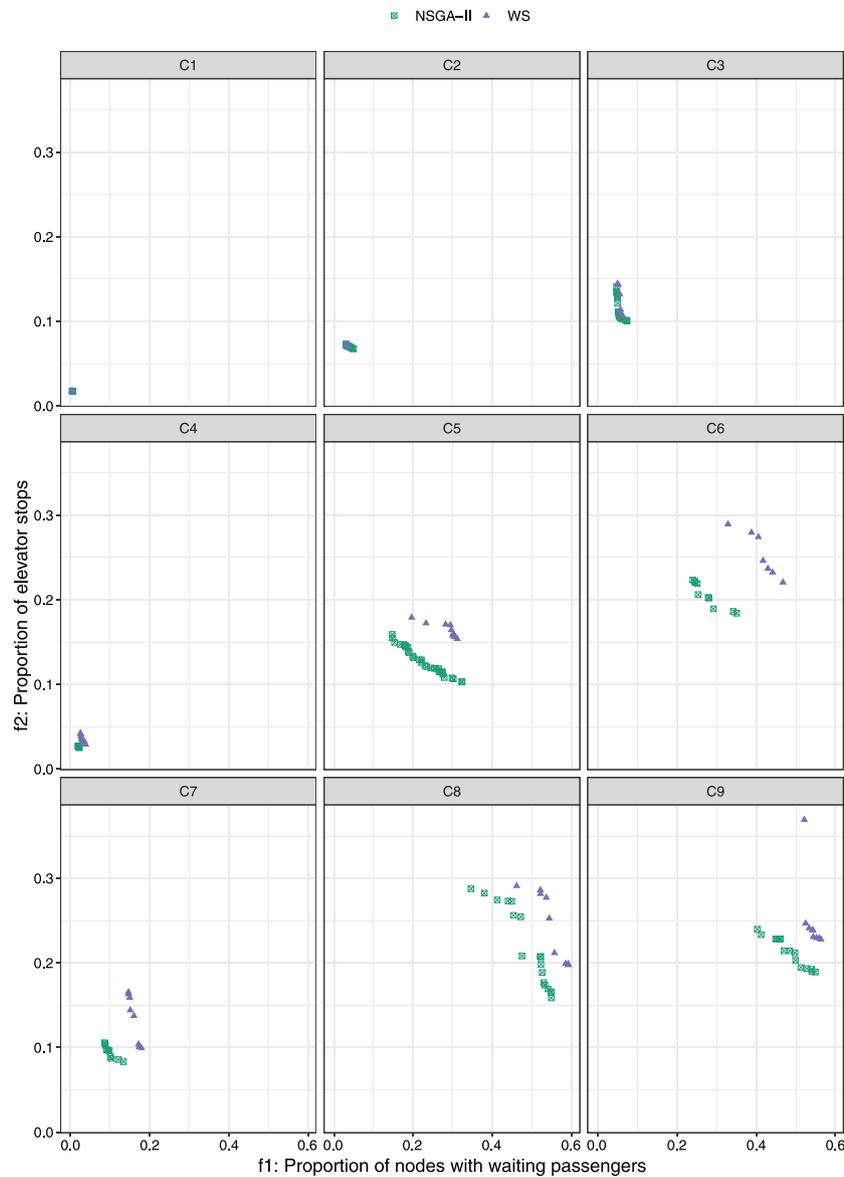


Fig. 6. Pareto front approximations resulting from typical runs of NSGA-II and WS. The first row shows the results for C1–C3, the second row for C4–C6, and the third row for C7–C9.

problem was formulated as a CMOP involving two conflicting objectives that both have to be minimized, i.e., the proportion of nodes with waiting passengers and the proportion of elevator stops, and the constraint on the maximum number of elevator skips. The objectives were normalized to compare the results over elevator systems with different numbers of floors and elevator cars. The S-Ring computational model of an elevator system was used as a prerequisite for numerical optimization. In contrast to most studies in this domain, we exercised the true multi-objective optimization approach that returns approximations of Pareto-optimal solutions to the problem. Five widely used MOAs, namely MTS, MOPSO, NSGA-II, DEMO, and MOEA/D, were deployed for this purpose and tested on nine test elevator system configurations of various complexity.

The experimental evaluation included the algorithm parameter tuning, systematic experiments on the test elevator configurations, and detailed analysis of the results. The more complex the elevator configuration, the more evident becomes the superiority of MOEAs over non-evolutionary algorithms. While the computational efficiency of the tested MOEAs is comparable as it mainly

depends on the cost of the S-Ring simulation, in terms of the effectiveness, NSGA-II performs best on average, and its parameter setting was found the most robust in the tuning process.

From the application point of view, the methodology represents a valuable tool, and the results offer new insights into the problem domain to decision-makers involved in the elevator system configuring and EGC design. The identified sets of nondominated solutions allow for trading between the objective values and the analysis of correlations between the objectives and the constraint for a better understanding of the problem.

The key directions of our further work in this domain include analyzing the produced trade-off control policies in the design space and enhancing the methodology for applications in advanced real-world elevator systems. Of particular interest in the latter case will be dynamically changing operating conditions of the elevator systems where, unlike in the current study, the systems performance will be optimized over different traffic situations.

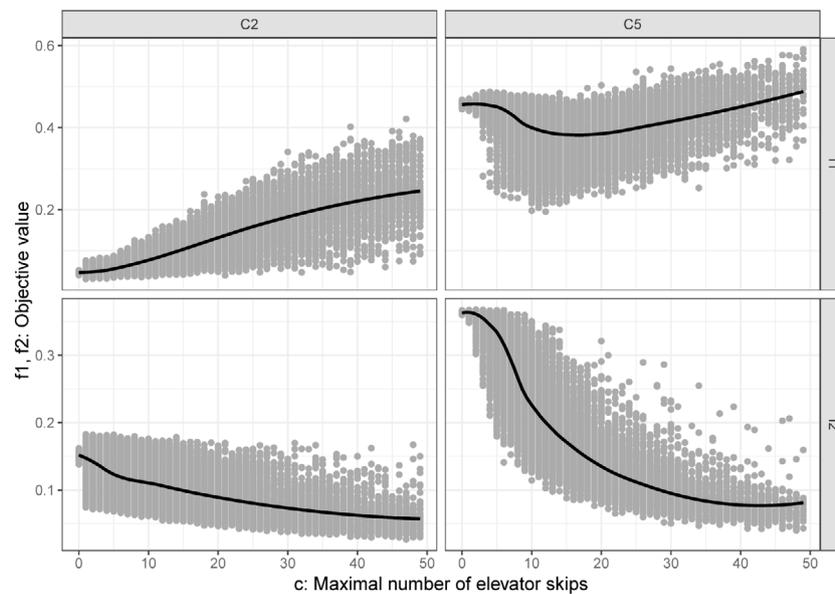


Fig. 7. Correlations between the objective and constraint values for elevator system configurations C2 (left) and C5 (right).

CRedit authorship contribution statement

Aljoša Vodopija: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data curation, Writing – original draft, Writing – review & editing, Visualization. **Jörg Stork:** Conceptualization, Methodology, Software, Validation, Formal analysis, Writing – original draft, Visualization. **Thomas Bartz-Beielstein:** Conceptualization, Methodology, Validation, Formal analysis, Writing – original draft, Supervision, Funding acquisition. **Bogdan Filipič:** Conceptualization, Methodology, Validation, Formal analysis, Writing – original draft, Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is part of a project that has received funding from the European Union's Horizon 2020 research and innovation program under Grant Agreement no. 692286. We acknowledge financial support from the Slovenian Research Agency (young researcher program and research core funding no. P2-0209). The work is also part of the Slovenian–German research collaboration supported by the Slovenian Research Agency (project BI-DE/20-21-019) and the German Academic Exchange Service (project 57515062). T. Bartz-Beielstein acknowledges support from the *Ministerium für Kultur und Wissenschaft des Landes Nordrhein-Westfalen* in the funding program *FH Zeit für Forschung* under the grant number 005-1703-0011 (project OWOS) and support from the *German Federal Ministry of Education and Research* in the funding program *Forschung an Fachhochschulen* under the grant number 13FH0071B6 (project KOARCH).

References

- [1] T. Bartz-Beielstein, M. Preuss, S. Markon, Validation and optimization of an elevator simulation model with modern search heuristics, in: T. Ibaraki, K. Nonobe, M. Yagiura (Eds.), *Metaheuristics: Progress as Real Problem Solvers*, Springer, Berlin, Heidelberg, New York, 2005, pp. 109–128, http://dx.doi.org/10.1007/0-387-25383-1_5.
- [2] A. Vodopija, J. Stork, T. Bartz-Beielstein, B. Filipič, Model-based multiobjective optimization of elevator group control, in: B. Filipič, T. Bartz-Beielstein (Eds.), *International Conference on High-Performance Optimization in Industry, HPOI 2018: Proceedings of the 21st International Multiconference Information Society, IS 2018*, Jožef Stefan Institute, Ljubljana, Slovenia, 2018, pp. 43–46.
- [3] S. Markon, A solvable simplified model for elevator group control studies, in: 2015 IEEE 4th Global Conference on Consumer Electronics, GCCE, IEEE, 2015, pp. 56–60, <http://dx.doi.org/10.1109/GCCE.2015.7398739>.
- [4] H.M. Hakonen, A. Rong, R. Lahdelma, Multiobjective optimization in elevator group control, in: P. Neittaanmäki, T. Rossi, S. Korotov, E. Oñate, J. Périaux, D. Knörzer (Eds.), *European Congress on Computational Methods in Applied Sciences and Engineering, ECCOMAS 2004*, 2004, p. 17.
- [5] Y.G. Sahin, S. Uzunbayir, B. Akcay, E. Yildiz, Real-time monitoring of elevators to reduce redundant stops and energy wastage, in: *Proceedings of the 2013 International Conference on Systems, Control and Informatics*, 2013, pp. 264–269.
- [6] T. Tyni, J. Ylinen, Evolutionary bi-objective optimisation in the elevator car routing problem, *European J. Oper. Res.* 169 (2006) 960–977, <http://dx.doi.org/10.1016/j.ejor.2004.08.027>.
- [7] T. Mahmud, S.M. Saiduzzaman, S.S. Ahabab, S.S. Saud Nikor, A simplified, novel and efficient approach to operate a group of elevators using a common control, in: 2019 IEEE International Conference on Robotics, Automation, Artificial-Intelligence and Internet-of-Things, RAAICON, IEEE, 2019, pp. 131–134, <http://dx.doi.org/10.1109/RAAICON48939.2019.56>.
- [8] P.E. Utgoff, M.E. Connell, Real-time combinatorial optimization for elevator group dispatching, *IEEE Trans. Syst. Man Cybern. A* 42 (1) (2012) 130–146, <http://dx.doi.org/10.1109/TSMCA.2011.2157134>.
- [9] A. Fujino, T. Tobita, K. Segawa, K. Yoneda, A. Togawa, An elevator group control system with floor-attribute control method and system optimization using genetic algorithms, *IEEE Trans. Ind. Electron.* 44 (4) (1997) 546–552, <http://dx.doi.org/10.1109/41.605632>.
- [10] D.L. Pepyne, C.G. Cassandras, Optimal dispatching control for elevator systems during uppeak traffic, *IEEE Trans. Control Syst. Technol.* 5 (6) (1997) 629–643, <http://dx.doi.org/10.1109/87.641406>.
- [11] J.-M. Kuusinen, M. Ruokokoski, J. Sorsa, M.-L. Siikonen, Linear programming formulation of the elevator trip origin-destination matrix estimation problem, in: B. Vitoriano, F. Valente (Eds.), *Proceedings of the 2nd International Conference on Operations Research and Enterprise Systems*, in: ICORES 13, 2013, pp. 298–303, <http://dx.doi.org/10.5220/0004338502980303>.
- [12] T. Miyamoto, S. Yamaguchi, McESim: A multi-car elevator simulator, *IEICE Trans. Fundam.* E91 (11) (2008) 3207–3214, <http://dx.doi.org/10.1093/ietfec/e91-a.11.3207>.
- [13] S. Markon, D.V. Arnold, T. Bäck, T. Beielstein, H.-G. Beyer, Thresholding—A selection operator for noisy ES, in: *Proceedings 2001 Congress on Evolutionary Computation, CEC, IEEE, Piscataway NJ*, 2001, pp. 465–472, <http://dx.doi.org/10.1109/CEC.2001.934428>.
- [14] T. Bartz-Beielstein, K.E. Parsopoulos, M.N. Vrahatis, Design and analysis of optimization algorithms using computational statistics, *Appl. Numer. Anal. Comput. Math.* 1 (2) (2004) 413–433, <http://dx.doi.org/10.1002/anac.200410007>.

- [15] R.A. Howard, *Dynamic Programming and Markov Processes*, MIT Press, Cambridge, 1960.
- [16] A. Onat, S. Markon, H. Kita, Y. Nishikawa, *A Benchmark Problem for Evaluating Reinforcement Learning Methods*, Tech. rep., 2002.
- [17] M. Ruokokoski, J. Sorsa, M.-L. Siikonen, H. Ehtamo, Assignment formulation for the Elevator Dispatching Problem with destination control and its performance analysis, *European J. Oper. Res.* 252 (2) (2016) 397–406, <http://dx.doi.org/10.1016/j.ejor.2016.01.019>.
- [18] N. Takahashi, S. Markon, A. Onat, Multi-objective optimization of the design of an elevator linear motor, in: 2013 IEEE Power Energy Society General Meeting, IEEE, 2013, pp. 1–5, <http://dx.doi.org/10.1109/PESMG.2013.6672412>.
- [19] T. Ishii, Elevators for skyscrapers, *IEEE Spectr.* 31 (9) (1994) 42–46, <http://dx.doi.org/10.1109/6.309960>.
- [20] S. Markon, Y. Komatsu, A. Yamanaka, A. Onat, E. Kazan, Linear motor coils as brake actuators for multi-car elevators, in: 2007 International Conference on Electrical Machines and Systems, ICEMS, IEEE, 2007, pp. 1492–1495.
- [21] Y. Okamoto, N. Takahashi, Minimization of driving force ripple of linear motor for rope-less elevator using topology optimization technique, *J. Mater. Process. Technol.* 181 (1) (2007) 131–135, <http://dx.doi.org/10.1016/j.jmatprotec.2006.03.057>.
- [22] A. Onat, E. Kazan, N. Takahashi, D. Miyagi, Y. Komatsu, S. Markon, Design and implementation of a linear motor for multicar elevators, *IEEE/ASME Trans. Mechatronics* 15 (5) (2010) 685–693, <http://dx.doi.org/10.1109/TMECH.2009.2031815>.
- [23] C. Gurbuz, E. Kazan, A. Onat, S. Markon, Linear motor for multi-car elevators, design and position measurement, *Turk. J. Electr. Eng. Comput. Sci.* 19 (6) (2011) 827–838, <http://dx.doi.org/10.3906/elk-1007-594>.
- [24] S. Markon, H. Kita, H. Kise, T. Bartz-Beielstein (Eds.), *Control of Traffic Systems in Buildings*, Springer, Berlin, Heidelberg, New York, 2006.
- [25] L.-Y. Tseng, C. Chen, Multiple trajectory search for unconstrained/constrained multi-objective optimization, in: 2009 IEEE Congress on Evolutionary Computation, CEC, 2009, pp. 1951–1958, <http://dx.doi.org/10.1109/CEC.2009.4983179>.
- [26] C.R. Raquel, P.C. Naval, An effective use of crowding distance in multi-objective particle swarm optimization, in: Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation, GECCO '05, 2005, pp. 257–264, <http://dx.doi.org/10.1145/1068009.1068047>.
- [27] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE Trans. Evol. Comput.* 6 (2) (2002) 182–197, <http://dx.doi.org/10.1109/4235.996017>.
- [28] T. Robič, B. Filipič, DEMO: Differential evolution for multiobjective optimization, in: C.A. Coello Coello, A. Hernández Aguirre, E. Zitzler (Eds.), Proceedings of the Third International Conference on Evolutionary Multi-Criterion Optimization, EMO 2005, in: Lecture Notes in Computer Science, vol. 3410, Springer, Berlin, Heidelberg, 2005, pp. 520–533, http://dx.doi.org/10.1007/978-3-540-31880-4_36.
- [29] Q. Zhang, H. Li, MOEA/D: A multiobjective evolutionary algorithm based on decomposition, *IEEE Trans. Evol. Comput.* 11 (6) (2007) 712–731, <http://dx.doi.org/10.1109/TEVC.2007.892759>.
- [30] F. Campelo, L.S. Batista, C. Aranha, The MOEADr package: A component-based framework for multiobjective evolutionary algorithms based on decomposition, *J. Stat. Softw.* 92 (6) (2020) 1–39, <http://dx.doi.org/10.18637/jss.v092.i06>.
- [31] T. Bartz-Beielstein, C.W. Lasarczyk, M. Preuss, Sequential parameter optimization, in: 2005 IEEE Congress on Evolutionary Computation, CEC, IEEE, 2005, pp. 773–780, <http://dx.doi.org/10.1109/CEC.2005.1554761>.
- [32] R Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria, 2013, URL <http://www.R-project.org/>.
- [33] A. Vodopija, Egccmop: Elevator group control as a constrained multiobjective optimization problem, 2020, URL <https://github.com/vodopijaaljosa/egccmop>.
- [34] GNU General Public License, version 3. URL <http://www.gnu.org/licenses/gpl.html>.
- [35] K.V. Price, R.M. Storn, J.A. Lampinen, *Differential Evolution*, in: Natural Computing Series, Springer, Berlin, Heidelberg, 2005, <http://dx.doi.org/10.1007/3-540-31306-0>.