Journal of Intelligent Manufacturing (2024) 35:3235–3257
https://doi.org/10.1007/s10845-023-02191-4

Evolutionary algorithms for a simheuristic optimization of the product-service system design

Henri Meeß[1] · Michael Herzog[2,3] · Enes Alp[3] · Bernd Kuhlenkötter[2,3]

Received: 25 February 2023 / Accepted: 1 August 2023 / Published online: 9 September 2023
© The Author(s) 2023

Abstract
Offering Product-Service Systems (PSS) becomes an established strategy for companies to increase the provided customer value and ensure their competitiveness. Designing PSS business models, however, remains a major challenge. One reason for this is the fact that PSS business models are characterized by a long-term nature. Decisions made in the development phase must take into account possible scenarios in the operational phase. Risks must already be anticipated in this phase and mitigated with appropriate measures. Another reason for the design phase being a major challenge is the size of the solution space for a possible business model. Developers are faced with a multitude of possible business models and have the challenge of selecting the best one. In this article, a simheuristic optimization approach is developed to test and evaluate PSS business models in the design phase in order to select the best business model configuration beforehand. For optimization, a proprietary evolutionary algorithm is developed and tested. The results validate the suitability of the approach for the design phase and the quality of the algorithm for achieving good results. This could even be transferred to already established PSS.

Keywords Product-Service Systems (PSS) · Business model prototyping · Simheuristics · Evolutionary algorithms · Metaheuristic optimization

Introduction

Due to globalization and high competition, manufacturers transform their strategic alignment and offerings by focusing on the customers' needs. As a result, the product portfolio gets expanded by combining products and services to Product-Service Systems (PSS) with the aim of providing a greater customer value (Moro et al., 2021). PSS are "*characterized by the integrated and mutually determined planning, development, provision and use of product and services shares*" (Meier et al., 2010). They enable producers to introduce innovative business models such as use-oriented or result-oriented. These business models focus on the benefits generated by the product rather than on the product itself. According to this the availability of a technical asset is offered instead of the ownership. Providers conduct the necessary services in order to realize the promised value proposition. On the other side, customers pay e.g., a monthly rate when the promised availability of the product is ensured instead of making a large investment upfront. Consequently, the majority of risks, e.g., technical risks are transferred from the customers to the providers (Reim et al., 2015). This way, manufacturers try to capture unique competitive advantages by generating higher revenues and long-term relationships with the customer (Li et al., 2020). Thus, in recent years, many providers of conventional business models have started to establish customer benefit-generating offerings in the form of PSS. Typical examples that can be noticed in daily life are car- and bike-sharing models (Köster, 2015).

Besides these advantages, providers face particular challenges when designing and developing PSS. This is because

✉ Henri Meeß
henri.meess@ivi.fraunhofer.de

Michael Herzog
herzog@lps.rub.de

Enes Alp
alp@lps.rub.de

Bernd Kuhlenkötter
kuhlenkoetter@lps.rub.de

1 Fraunhofer Institute for Transportation and Infrastructure Systems IVI, Ingolstadt, Germany

2 Center for the Engineering of Smart Product-Service Systems (ZESS), Ruhr-Universität Bochum, Bochum, Germany

3 Chair of Production Systems (LPS), Ruhr-Universität Bochum, Bochum, Germany

not only the products or the combined services must be optimized, but rather the combination of the two must create a higher customer benefit than the individual components. This requires detailed knowledge from various domains and significantly increases the complexity of the optimization process (Boßlau, 2021). This complexity of implementing PSS and the limited experience with similar business models have presented challenges in generating profits (Yan et al., 2021). Consequently, despite the extensive development of various approaches in the literature, the projected widespread adoption of PSS made in 2010 (Meier et al., 2010) has not yet been achieved. A comprehensive review of these methodologies reveals the necessity for novel approaches to design successful PSS. Recognizing the research gap, the authors identify the integration of data science and engineering design techniques as a promising avenue recommending their combination to develop approaches that aim to achieve the optimal configurations of PSS in given cases (Brissaud et al., 2022).

In alignment with these insights, the central hypothesis of the presented work is that the profitability of PSS depends mainly on the customer-oriented design and configuration of the business model. Thereby, the profitability of the provider is not the only target value in the development of PSS business models. The satisfaction of the customers has also to be considered in the business model configuration in order to grant long-term success (Köster, 2015). To find the optimum between these two concurring targets new quantitative methods are needed. The presented work links data science techniques with methods of engineering design to develop a new approach to the design of PSS and contributes to this research gap presented in (Brissaud et al., 2022).

To assess the achievement of these objectives by each business model configuration, the use of computer simulations has proven to be a valid methodology (Boßlau, 2021). The determination of the optimal PSS business model configuration, however, is a highly complex task since the solution space is huge and each decision has to be aligned with the others in order to reach the optimum of the conflicting objectives profitability for the provider and the satisfaction of the customers (Köster, 2015). One way to solve such combinatorial optimization problems is the use of metaheuristic optimization. Metaheuristics are a subset of stochastic optimization methods that primarily rely on the utilization of random streams within the search mechanisms (Chih, 2023). Metaheuristic optimization is applied to problems where the solution space is so large that calculating every single possible solution takes too long and is too costly. Mostly inspired by natural phenomena, metaheuristic algorithms provide optimal or good enough solutions in a short time (Abualigah et al., 2022). Combining metaheuristics with simulations to a simheuristic approach enables considering uncertainties (Juan et al., 2015) which are immanent in the design phase due to the long-term character of PSS and the unpredictability of the future (Keine gen. Schulte & Steven 2012). Hence, this paper contributes to the design phase of PSS a new quantitative approach, that combines metaheuristic optimization algorithms with stochastic simulations within a simheuristic framework in order to optimize the PSS business model configuration upfront or even after deployment. The main focus of this paper is on the development and design of this optimization algorithm by following a two-step approach. In the first step, a series of sub-studies and experiments are conducted to create important insights into the applicability of evolutionary algorithms (EA) in the PSS domain. In addition, different required adaptation technics are investigated. In the second step, based on these findings a highly efficient optimization algorithm is developed, applied, and validated to be suitable for optimizing PSS business model configurations within a simheuristic framework during the design phase. In contrast to existing qualitative approaches, the presented approach promises optimized and quantitatively evaluated business model configurations.

The rest of the paper is structured as follows. First, the necessary foundations are laid in the Sect. "Foundations". For this, the challenges in the decision-making during the design and development of PSS are outlined. Based on this, the use of simulations and heuristic methods as well as the combination of both as a simheuristic are derived as a potential solution approach. In the Sect. "Sub-study", various properties of optimization algorithms are examined and tested regarding their quality to derive requirements and insights for the development of a combined algorithm. The Sect. "Main-study" presents the developed optimization algorithm and validates its performance by applying it to a conceptual use-case. The results and findings are discussed in the Sect. "Discussion". Finally, this paper finishes with conclusions and an outlook for future work.

Foundations

Design and development of Product-Service Systems

The design and development of PSS is a complex process whereby the results of the decisions in this phase are only revealed in retrospect and can only then be evaluated. Due to the long-term character of PSS business models, the return on investment lies further in the future than in classical product development. Thus, poor decisions in the early stages of the design phase can lead to negative outcomes like decreased customer satisfaction, lower sales, and increased costs (Keine gen. Schulte and Steven 2012). In a recent keynote, Brissaud et al. (2022) point out that the complexity of the design phase is one of the main reasons why PSS are not as well spread in the industry as expected ten years ago.

S1: Transition to Alternative Realizations Options	S6: Distribution of Risks
S2: Distribution of PSS Objects	S7: Revenue Models
S3: Numerical Variation of the Necessary PSS Objects	S8: Variation of Property Rights
S4: Level of User-Integration	S9: Spatial Distribution
S5: Use of External Factors	

Fig. 1 Variation strategies for PSS configurations according to Köster (2015)

The mentioned complexity in the design and development phase is caused by a large number of decisions and their range of definitions. The decision-making process itself is associated with a high degree of uncertainty due to the unpredictability of the future. These challenges of the design and development of PSS can be illustrated by the *variation strategies* (see Fig. 1) identified by Köster (2015). These strategies are superordinate design decisions and by no means details relating to the integrated service or product.

Based on these variation strategies, a car-sharing provider could be confronted with the following questions and decisions (see Fig. 2). Applying S1, a provider could think of the integration of autonomous driving (in the future) as an alternative for driving by the customer. Regarding S2, the provider could vary the areas served by the business model. Referring to S3, a provider could vary the number of its employees, vehicles, or parking spaces. With S4, the provider could make decisions about the fueling policy. He could set incentives for the customers who fuel the rented cars after driving. Deploying S5, the provider could enter into partnerships with park space providers. Referring to S6, the provider could vary the shares of the risks and decide if the customer is liable for damages. Concerning S7, the provider could vary the revenue models in such a way, that the customer would pay, for example, for the minute or the kilometer, etc.

Considering these variation strategies, it gets evident that the solution space in the design phase of a PSS is huge, and finding the perfect match between provider objectives and customer needs seems impossible. Every decision in one area of the PSS and its business model is linked to a decision in another area (Osterwalder & Pigneur, 2013). Increasing the number of vehicles and widening the distribution of the served area could lead to higher customer satisfaction, however, would also lead to higher costs and other challenges e.g., ensuring the cleanliness of all vehicles. If the customer will be integrated into the cleaning process, adjustments in the revenue models must be made. Determining the revenue model as Pay-per-minute could lead to faster-driving customers which could result in careless driving or even accidents. Thus, the provider must also adjust the risk distribution and decide who will be liable for which damages.

These explanations lead to the conclusion that achieving an optimal configuration of a PSS, where all decisions are aligned with each other and, at the same time, the profitability of the business model can be predicted is impossible without appropriate methodological support. This finding is in line with the results of the extensive research of Brissaud et al. (2022). After analyzing existing design methodologies, the authors argue for linking data science methods with engineering design and present the lack of global optimization methods as a research gap.

To optimize a PSS configuration, different configuration options must be evaluated according to the desired objective e.g., minimum costs, maximum customer satisfaction, etc., or a combination of them. For this task, quantitative methods are necessary that can deal with the unpredictability of the future and thus the real behavior of PSS system components, customers, and the environment during the operation phase. To reduce the uncertainties and risks in this respect a realistic simulation of the PSS environment is an obvious option (Wrasse et al., 2015).

Simulation of Product-Service Systems

Computer simulations can be used as decision support systems for the PSS design and development phase as well as for the operations phase of PSS as a controlling instrument. The advantage of simulations in comparison to traditional methods (e.g., the Business Model Canvas in (Osterwalder & Pigneur, 2013) is that designers and developers receive quantitative feedback for their decisions which enables them to rate and compare these decisions objectively (Boßlau, 2021). In the context of PSS, simulations were used in various issues and on different abstraction levels. Kimita et al. (2012) use simulations to estimate the costs of service processes. Medini et al. (2015) present a simulation-based approach for the evaluation of PSS value chain configurations. Boßlau (2015) models and simulates PSS business models as a whole. Thereby, it is investigated how different risk scenarios affect the profits of the PSS provider. Alice et al. (2018) use simulations for the assessment of service delivery processes in the context of PSS. Alp et al. (2021) evaluate the use of remote assistance in service delivery processes.

Musa et al. (2017) elaborate that the typical modeling paradigms for the simulation are Discrete Event Simulation

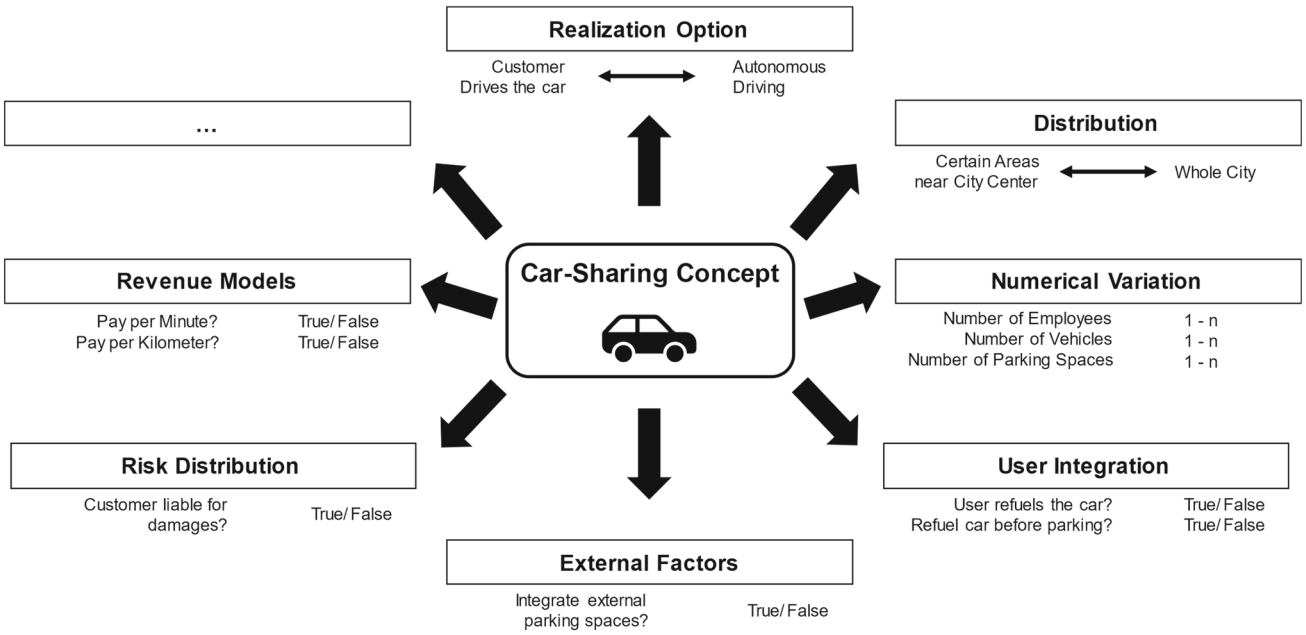

Fig. 2 Variation strategies for a car-sharing concept

(DES), System Dynamics (SD), and Agent-based Modeling (ABM). DES is characterized by a sequence of discrete states in which the operations or state transitions are triggered by events (Wrasse et al., 2015). SD is a modeling paradigm for a holistic analysis of complex and dynamic systems. Using SD, the system is viewed from a top-down perspective. The detailed behavior of single elements is not considered. Rather, it attempts to model the abstract system behavior by identifying reinforcing or balancing loops in the system (Bala et al., 2017). In ABM, agents act as distinctive objects that operate in the simulated environment according to certain rules (Helbing, 2012). Due to the complexity of PSS as a sociotechnical system, Alp et al. (2023) argue for the use of a multimethod approach, the combination of all three methods, for the modeling and simulation of PSS business models in order to achieve a better reflection of reality.

Heuristic optimization

For the optimization of many real-world problems, an exhaustive search, where all possible solutions are evaluated is not applicable, since the solution space is too large. Here, optimization "*can be thought of as the process of attempting to find the best possible solution amongst all those available*" (Burke & Kendall, 2014). One way to cope with these kinds of problems is the use of heuristic techniques. Heuristics can be defined as "*a technique which seeks good (i.e., near-optimal) solutions at a reasonable computational cost without being able to guarantee […] optimality*" (Andrew, 1998). Better results can be achieved by using metaheuristics. Metaheuristics refer *"to a master strategy that guides and modifies other heuristics to produce solutions beyond those that are normally generated in a quest for local optimality*" (Glover & Laguna, 1999). The comprehensive review of Abualigah et al. (2022) shows the application of these techniques for design problems in the area of mechanical engineering and proves their success in optimizing real-world problems. Most of the classical metaheuristics are inspired by phenomena in nature (Dokeroglu et al., 2019) such as the *colony search optimization algorithm* (Wen et al., 2022), *whale optimization algorithm*, *artificial bee colony algorithm*, *cuckoo search*, *firefly algorithm*, or the *gray wolf optimizer* or such as evolutionary algorithms (Abualigah et al., 2022). The inspiration for EAs lies in the behavior and structure of living organisms during biological evolution (Zhang et al., 2011). The main components contained within an EA are listed and briefly explained below in Fig. 3.

A general procedure for EAs is shown in Fig. 4. There, a simplification of the real procedure is visible. Due to the complexity and freedom in the design of EAs as well as the variability of possible tasks, different strategies are applied in practice to achieve the best possible optimization. However, most of these can be traced back to slightly modified forms of the general sequence from Fig. 4.

Furthermore, two main branches of EAs can be distinguished. On the one hand, genetic algorithms (GA) are specially designed for optimization within discrete solution spaces (Mitchell, 1997; Spears et al., 1993; Whitley, 2001). On the other hand, evolution strategies (ES) usually perform optimizations on continuous solution spaces (Beyer & Schwefel, 2002). Both types of algorithms come with their own methods regarding the generations of offspring and

Fig. 3 Main Components of EAs (Weicker & Weicker, 2003)

Individual	Is discretely characterized by the expression of a genotype and is therefore represented by a vector of parameters which, analogous to DNA in biology, determines its real expression.
Population	The population is the totality of individuals that exists simultaneously within the EA.
Offspring	Individuals in the current population produce children by mutation and/or recombination, who, depending on the EA, enter the next generation of the population.
Fitness	The fitness or objective function defines the quality of an individual's adaptation to its environment. It is a direct and quantitative measure of how well an individual fulfils a given objective.
Selection	The decision as to which individuals will pass into the next generation and which parents will be considered for the generation of descendants is made based on fitness or by chance.

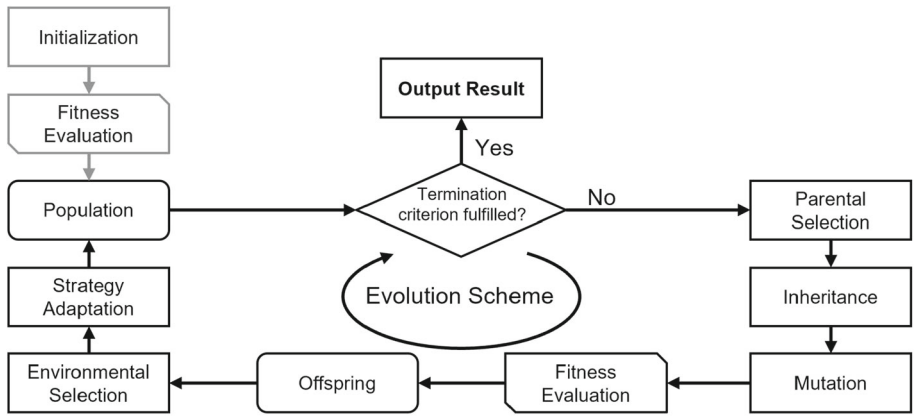

Fig. 4 Simplified schematic representation of an optimization using EAs (Weicker & Weicker, 2003)

strategy adaptations. Regardless, the basics of environmental selection are common for both. For this purpose, two strategies have usually to be considered. First, the comma selection [e.g. (μ,λ)-GA] where the new generation (here, λ denotes the number of *offspring generated per generation*) is formed solely out of the best newly generated offspring (Beyer & Schwefel, 2002; Glasmachers, 2013; Spears et al., 1993). Second, the plus selection [e.g. (μ + λ)-GA] whereby the next generation consists of the best individuals of the totality of the prior generation (here, μ denotes the number of *individuals within a population*) and the generated offspring (Bäck & Schwefel, 1993; Whitley, 2001).

Based on these building blocks, various algorithms for diverse problems can be generated by sensible selection of the parameters, offering different advantages and disadvantages. The particularly used algorithms and varied building blocks as well as the incorporated strategies for mutation and inheritance are introduced briefly in the appropriate sections (compare Sect. "General suitability of the different algorithms").

Generalizability and theoretical motivation

In principle, a simulative/theoretical study can naturally not be transferred 1 to 1 to real-world applications, however, EAs show very good properties in terms of invariance and generalizability. Those invariant properties allow the transfer of different characteristics when solving benchmark problems to optimization tasks of a similar type. Depending on the type of EA, the algorithm is invariant under strictly monotonous transformations of the fitness function as well as rotations and translations of the search space, if the initial search points are changed identically. Besides, scale invariance and invariance against invertible linear transformations of the search space, with corresponding modifications to start points and covariance matrix, can be achieved (Glasmachers, 2017; Hansen, 2006; Mersch et al., 2006).

Further research demonstrates additional invariance properties in different application areas of EAs (Salimans et al., 2017). Those promising characteristics of EA motivate the initial general investigation of different EA strategies for the optimization of PSS, as the stated generalizability indicates a reasonable degree of transferability to dedicated already implemented or proposed PSS.

Simheuristics

The simheuristic approach integrates "*simulation and optimization techniques to efficiently tackle a combinatorial optimization problem characterized by stochastic components*" (Gok et al., 2020). By this extension, metaheuristics, which are predominantly designed for solving deterministic problems, can additionally be used for coping with optimization problems under uncertainty. Thus, simheuristics allow the consideration of e.g., random customer demands while optimizing the problem towards a given objective. In contrast

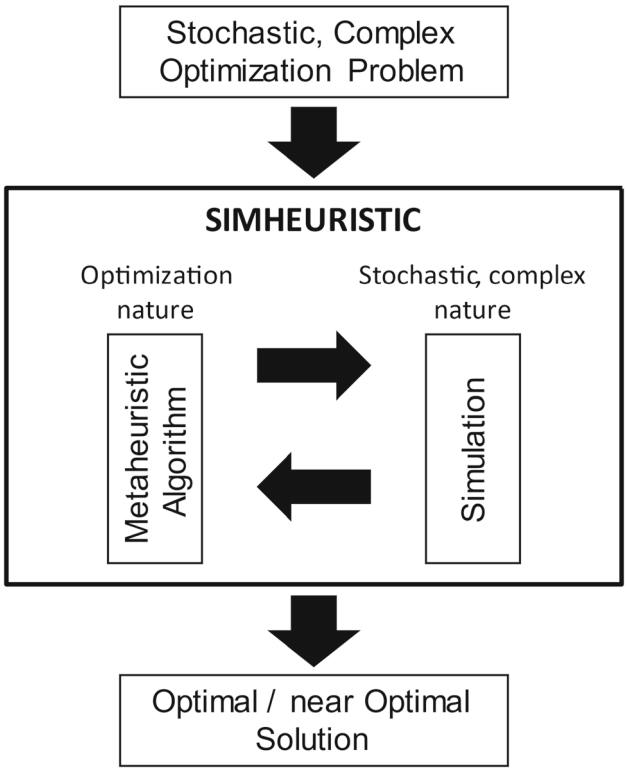

Fig. 5 Framework of the simheuristic approach adopted from Juan et al. (2015)

to the use of analytical models, the integration of validated simulation models enables a better representation of the complexity of the problem (Chica et al., 2017).

A general framework for the simheuristic approach is visualized in Fig. 5. The complexity and uncertainty of a decision are transformed into an optimization problem whereby a simulation model represents the stochastic and complex nature of the problem, and the metaheuristic algorithm is used for its optimization with the aim of obtaining an optimal solution. In detail, the metaheuristic algorithm generates solutions, that serve as input for the stochastic simulation. The simulation is first run with a low number of replications to estimate the quality and feasibility of the solutions and rank them regarding their proximity to the optimization target. The best solutions are then used to run a large number of simulations, analyze their outputs, and select the best solution (Juan et al., 2015).

The application fields of the simheuristic approach are manifold. Calvet et al. (2016) propose the combination of metaheuristics with simulation for solving distributed scheduling problems with stochastic processing times. The simheuristic approach optimizes starting times of each task while minimizing completion time or accumulated deviations with respect to the deadline (Calvet et al., 2016). Reyes-Rubiano et al. (2017) propose a simheuristic approach for solving the capacitated vehicle routing problem, considering sustainability indicators and stochastic traveling times. Through computational experiments, the authors demonstrate the ability of the approach to provide high-quality solutions within a short computational time (Reyes-Rubiano et al., 2017). Wang et al. (2018) present a simheuristic approach for optimizing the allocation and scheduling of reversible lanes at a container terminal gate. The objective of their approach is to improve service level and maximize resource utilization while considering the constraints of limited space and imbalance traffic volumes (Wang et al., 2018). Panadero et al. (2020) consider the routing of drones in a team being utilized for surveillance. The authors propose a simheuristic approach to solve the stochastic team orienteering problem (Panadero et al., 2020). Bi et al. (2020) present a two-level principal-agent model for controlling the schedule risk of IT outsourcing projects. The model utilizes a genetic algorithm based simheuristic approach for providing decision support to scheduling IT outsourcing projects (Bi et al. 2020). Yazdani et al. (2020) use the simheuristic approach for optimizing evacuation strategies by determining the right number of buses and routes for picking up passengers. Caldeira and Gnanavelbabu (2021) developed a simheuristic approach for the flexible job shop problem with stochastic processing times. Their approach is capable of obtaining acceptable solutions for large-size problems in a short computational time (Caldeira & Gnanavelbabu, 2021). Yan et al. (2023) introduce a multi-objective model for fair relief material distribution during disasters using a hybrid metaheuristic algorithm. Through a case study on Covid-19 logistics in Wuhan, it demonstrates the algorithm's efficiency and competitiveness compared to other methods (Yan et al. 2023).

The wide range of applications of the simheuristic approach shows its strength for the optimization of complex, stochastic problems to which the design and development of PSS can be added. To use the simheuristic approach for the design and development of PSS, different algorithms must be tested for their suitability for the optimization of PSS. Further, the best-suited algorithms need to be adapted, further developed, and tested to be used in PSS design and development as an optimization tool in the future.

Design of a simheuristic approach for the optimization of PSS

The selection, adaption, and application of the right algorithm for the optimization of PSS business models during the design and development phase pose a great challenge. There are various reasons for this. One reason lies in the conversion of the PSS concept into a parameterized simulation model, which allows it to be manipulated by an optimization

algorithm. Thereby, the parameters that need to be varied during the optimization can be very heterogeneous. Besides discrete, direct parameters like e.g., the number of vehicles, indirect parameters like e.g., the revenue model, the type of service provision, and similar have to be translated into a form that can be considered by the optimization algorithm. In particular, the modeling of dependent and independent variables, which are defined in the PSS business model concept as decisions such as maintenance strategy dependent number of employees, in a way that optimization can take place, turns out to be a complex task. In addition to these exemplary challenges in the translation of the PSS business model concept to a simulation model, algorithm-specific components must also be further detailed during the development, adaptation, and application of an algorithm. In evolutionary algorithms, these are, for example, the size of the population or the selection strategy for evolving a new generation. Furthermore, the question arises of how the stochastic properties influence an evaluation of PSS business models in a fitness function and how this noise has to be taken into account when evaluating the performance of an algorithm.

Methodology

In this paper, a new EA-based optimization algorithm is developed based on the characteristics of the decisions in the design and development of PSS. The methodology for the approach is visualized in Fig. 6.

Due to the fact, that there are several interlinked challenges to the optimization of PSS concepts, we conduct six sub-studies in the first place. Within each sub-study, we highlight one challenging sub-aspect (listed in Fig. 6) and analyse, which strategies and characteristics of the individual components of the algorithm are best suited for the particular optimization challenge. Based on these insights a new optimization algorithm is synthesized. In the Sect. "Main-study", the design, application, and performance of the new algorithm are shown. This approach allows a significant reduction of complexity by considering the sub-issues individually. Without this individual consideration of the sub-issues, the complexity of the optimization of the algorithm itself would grow exponentially due to the combinatorial combination of different building blocks for each issue. Handling this is not feasible, especially against the background of the long simulation runtimes for the main. Additionally, a dedicated tuning of the hyperparameters would be necessary to optimize the optimization algorithm. This would further increase the complexity in the algorithm design search space and would make it impossible to realize an in-depth study on this topic with the available computational resources. Therefore, for the initial study on this topic, the first step is to investigate what independent insights can be gained concerning the individual sub-aspects. Then the results can be combined to demonstrate that the methods show certain generalizability and no incompatibilities or unpredictable degradations in the overall algorithm performance occur due to the combination of the individual findings. Therefore, a faster and easier-to-solve simulation setup is used for the sub-studies, in order to be able to carry out a larger number of examinations. By combining the findings within one optimized approach for the main-study, which has a significantly higher complexity, the generalizability of the individual findings and their combinability can then be verified.

Sub-study

As outlined, there are many challenges in implementing developing, adapting, and applying optimization algorithms on PSS configurations. In this Section, the following sub-aspects/research questions are further analysed in the context of PSS optimization:

1. How to deal with discrete validity ranges?
2. How to optimize setups with noisy fitness functions?
3. Which EA approaches are in general best suited as initial building blocks for the optimization of PSS?
4. How to optimize PSS setups with dependent variables/design decisions?
5. How to deal with various variable types during the optimization?
6. What is the best method to perform multi-objective optimization for PSS based on EA?

For the investigation of the sub-aspects, a simplified version of the Power-by-the-Hour PSS from Rolls Royce is modelled in a simulation. With Power-by-the-Hour, the aircraft turbine manufacturer Rolls-Royce established a performance-based business model in 1962 in which the value-adding activities of its customers were systematically taken over. Instead of the turbines, Rolls-Royce sells flight hours generated by their products. To this end, Rolls-Royce monitors the technical availability of the turbines using systematic monitoring of the turbine's health condition and performs the necessary maintenance required to fulfil the promise of customer value. This allows customers to focus on their core competencies and only pay for performance that adds value. (Friedrich, 2016; Köster, 2015; Smith, 2013).

The business model described above is transferred to a simulation environment. This simulation can be adapted by varying parameters like the number of aircraft, the number of service stations in different airports, and the number of technicians for each service station. Thus, the simulation

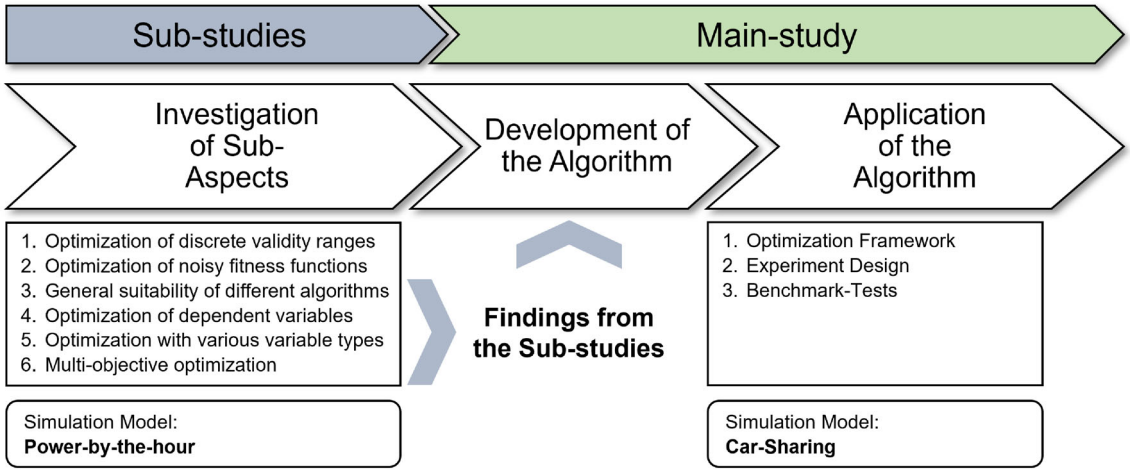

Fig. 6 Methodology for the development of a new optimization algorithm

model provides a 9-dimensional solution space with approx. $3.9 \cdot 10^{11}$ possible PSS configurations.[1] Considering all combinatorial solution possibilities for the sub-aspects is not feasible without further ado. Due to the huge size of the solution space and long simulation times for each fitness evaluation, finding the optimal configuration following the Brute-Force method (simulation of every single possibility) is not applicable.

To investigate the different sub-research questions, the optimization parameters of the simulation are changed or adapted to the respective objectives. The sub-research questions investigated in the sub-studies are briefly presented below and the investigated approaches are briefly explained. The overall findings of this first investigation stage to develop an optimization algorithm suitable to add significant benefit in the development of PSS are summarized all together at the end of this Section while further relevant plots can be found in the appendix.

Sub-study: optimization of discrete validity ranges

Considering real-world PSS setups, it is possible that the optimization parameters may only lie within a certain value range. Furthermore, no unrealistically large values should be considered within the optimization of PSS configurations. Therefore, it is necessary to observe the case of existing parameter constraints. Several ways to deal with such constraints include (Coello Coello, 2002; Michalewicz & Schoenauer, 1996; Rodrigues et al., 2014; Salcedo-Sanz, 2009; Woldesenbet et al., 2009):

– *Decoding of feasible solutions*
– *Resampling*
– *Repair of infeasible points*
– *Penalties*

Each of these methods can affect the optimization differently. Especially in combination with step size control the choice of the procedure and the style of implementation must be chosen reasonably. In real-world applications, it cannot be assumed that all restrictions are always well-known. For example, a combination of two parameters can generate a singularity within the simulation. In this case, neither a mapping to valid areas, a penalty at the level of the violation, nor a repair of the search points can be used. Therefore, at first, only relatively generally applicable methods should be considered. This includes the resampling of a search point and the death penalty. However, the latter can strongly influence the step size control and lead to a rapid convergence of the optimization if used incorrectly. Therefore, for the practical application due to the black-box representation and its possibly unknown constraints, the resampling method to deal with infeasible search points is primarily chosen.

Sub-study: optimization of noisy fitness functions

In general, it must always be assumed that PSS simulations do not exactly reflect the real behaviour of the business model. It is not possible to predict the individual behaviour of each customer, product, and service in detail. Random deviations in the functionality of the product, the behaviour of customers, or significant environmental influences (car sharing: for example, a road closure) influence the PSS. Therefore, in many cases, it makes sense to consider stochastic components when modelling the PSS. To achieve this, random events are

[1] The varied parameters and validity ranges for each parameter are listed for the different sub-studies in each case as a table together with the result plots—to be found in the Appendix.

modelled in the corresponding simulations. Thus, a search point does not behave identically in every fitness evaluation. Rather, two consecutive fitness calculations of the same point can result in very different fitness values. In this case, the search point may only be optimized for a very specific fixed sequence of events within the simulation and delivers significantly worse results faced with the smallest random deviations of the events. To deal with fitness noise generated this way, the re-evaluation method is chosen, in which the fitness evaluation is directly performed several times on different seeds and the results are averaged to generate the overall fitness. This method is used because it allows a stable and robust optimization to be expected and additionally guarantees the comparability of all investigated methods. It is also possible to dynamically adjust the number of re-evaluations to the search space and the level of fitness noise in order to reduce the required simulation budget (Hansen et al., 2009).

Sub-study: general suitability of the different algorithms

To rate the overall suitability of different EA configurations, first, a meaningful benchmark must be defined. Random Search (RS), gradient descent (GD), and the package *pycma* developed by Hansen et al. (2022) are therefore investigated in terms of their suitability to represent a competitive benchmark.

Random search: RS is a common benchmark for black-box problems with huge and inhomogeneous fitness landscapes. In addition, RS is the only meaningful baseline to compare different algorithms and approaches along a wide range of applications due to its general applicability and fairness in terms of tuned hyperparameters.

Gradient descent: GD is used to investigate, whether gradient-based methods are well suited to find meaningful results for the optimization of PSSs. Due to the black-box nature of the task, GD must manually compute the gradient from multiple fitness evaluations for each optimization step.

Pycma: The library *pycma* contains several implementations of Covariance-Matrix-Adaptation (CMA)-ES algorithms, which are designed for difficult optimization tasks in continuous search spaces. Therefore, this library is used as a further reference to the approaches presented in this work.

The performance results of the different benchmark approaches (compare Sect. "Summary of the findings from the presented sub-studies") are on the one hand used to understand the performance of different EA approaches and also compared to a valid baseline. On the other hand, the best benchmark will be defined as the reference approach for the main-study.

To evaluate the basic suitability of different algorithm types for the optimization of PSS and to get a general understanding of the optimization behavior of EA within the context of PSS optimization, the following general approaches are compared:

Basic (1+1)-ES: The (1+1)-ES algorithm is one of the simplest forms of EAs. Basically, a new search point $\boldsymbol{y}$ is determined from a search point $\boldsymbol{x}$ by mutation via a Gaussian distribution and a variable step size σ. In the case of gaining better fitness with the new point compared to the old one, it will be adopted—otherwise, $\boldsymbol{x}$ will remain the current search point. This property makes it a classical hill climber.

(1+1)-ES with step size fitting: This EA is also a (1+1)-ES. The main difference from the basic version is the sampling approach for search points. Since there are significant differences in the validity ranges of the individual parameters in the PSS used, the step size is adapted to the individual parameters. This is done by an elementwise multiplication of σ with a vector that contains the valid range of each parameter.

(1 + 1)-GA: The (1 + 1)-GA is the equivalent to the (1 + 1)-ES in the GA domain. Again, one new point $\boldsymbol{y}$ is generated based on a search point $\boldsymbol{x}$. However, this is done by a different mutation method. For each parameter with a certain probability, it is decided, whether its value is changed or not. This probability is defined by the mutation rate. At a mutation rate of 10%, every tenth parameter is mutated on average. If a parameter is mutated, the new value is randomly taken from a discrete uniform distribution $U(\boldsymbol{x_min}_d; \boldsymbol{x_max}_d)$ of the valid value range for the according dimension$d$. The selection of the old value is also permissible here. Afterwards, the fitness of the two search points is compared and the one with the higher fitness is stored for the next iteration. An adjustment of the mutation rate in contrast to the already introduced ESs does not take place.

(1 + 1)-GA: The (1 + 1)-GA is the equivalent to the (1 + 1)-ES in the GA domain. Again, one new point $\boldsymbol{y}$ is generated based on a search point $\boldsymbol{x}$. However, this is done by a different mutation method. For each parameter with a certain probability, it is decided, whether its value is changed or not. This probability is defined by the mutation rate. At a mutation rate of 10%, every tenth parameter is mutated on average. If a parameter is mutated, the new value is randomly taken from a discrete uniform distribution $U(\boldsymbol{x_min}_d; \boldsymbol{x_max}_d)$ of the valid value range for the according dimension$d$. The selection of the old value is also permissible here. Afterwards, the fitness of the two search points is compared and the one with the higher fitness is stored for the next iteration. An adjustment of the mutation rate in contrast to the already introduced ESs does not take place.

(1+1)-ES/GA: This algorithm is a hybrid between a conventional (1+1)-ES and a GA. The special characteristic of this algorithm is the type of mutation. The mutation is carried out via a hybrid approach of ES and GA where the ES part determines the interval around the parent value in each dimension and the sampling is done by the GA part with a uniform distribution within this interval.

(μ,λ)-GA: (μ,λ)-GAs have a similar structure to the (1+1)-GA. The main differences, however, are the number of individuals in the generation of parents and offspring and the additional use of crossover operators for inheritance. In addition, a comma selection is performed.

(μ,λ)-GA plus elitism: This algorithm corresponds in its basic structure to the (μ,λ)-GA. The only difference is that a weak form of elitism is used. For each group of offspring, the best individual from the parent generation is copied before any offspring are generated.

(μ+λ)-GA: As described in Sect. "Heuristic optimization"-, plus selection represents a strong form of elitism. This characteristic is the biggest difference to the (μ,λ)-GA. After the successful generation of the offspring, the population of the next generation (iteration) is formed from the altogether best individuals of the old population $\boldsymbol{X}$ and the offspring $\boldsymbol{Y}$.

(μ,λ)-ES with CMA: The class of (μ,λ)-ES with CMA used here is inspired by algorithms of (Hansen, 2006; Krause & Glasmachers, 2015). The biggest difference to the GAs is the inheritance strategy. The generation of the offspring population occurs through a Gaussian distribution centred around the centre of gravity of the previous population $\boldsymbol{X}$ and described by the optimized covariance matrix derived from the CMA.

(μ+λ)-ES with CMA: This algorithm combines the invariant properties of the CMA with the selection pressure of plus selection.

Sub-study: optimization of dependent variables

PSS configurations can be based on different sets of parameters where some parameters depend on the chosen overall strategy – which may also be subject to optimization (and therefore be represented as an optimization parameter). Therefore, an optimization of dependent variables and sets may be necessary. The two strategies that are examined to solve this issue effectively are explained below:

Loop—Here the PSS configuration types are optimized in loops over a fixed number of iterations. The parameters of the inactive configuration type remain unchanged. Afterwards, either the configuration type can be changed to optimize the other parameter set or a further loop can be started.

Holding—This procedure only changes those parameters that are assigned to the currently active configuration. All other parameters are transferred from one generation to the following without any modification. Compared to the loop strategy the parameter which determines the configuration type is also subject to optimization. Therefore, it can change during every evolution step for every new sample.

Sub-study: optimization with various variable types

PSS configuration parameters can be characterized by discrete as well as continuous variables. Traditionally EA focus on one of the two possibilities. To optimize PSS with an inhomogeneous variable configuration, several approaches are investigated. All algorithms used are based on the approaches already presented in the Section about general suitability and are extended by different strategies. Each of these strategies is applied to the (1 + 1)-ES and to the (μ + λ)-ES. Only the offspring generation is changed. Additionally, the population-based algorithms use CMA besides the online step size adaptation to adjust themselves to the solution space:

The first strategy (ES) applied is the simplest one possible, since it only rounds the parameters generated by the normal distribution and step size σ.
The second strategy (GA/ES) varies the real-valued parameters according to the rules of the ES and mutates the integer parameters according to the mutation rule of the (1+1)-GA. This represents a mixed form of the two basic EA approaches GA and ES. Although not all variables are changed simultaneously due to the characteristics of the GA, the integer variables might be mutated over the whole validity range.

The third strategy (ES/GA interval) is based on the mutation approach used in the algorithm (1+1)-ES/GA. The real-valued parameters continue to be mutated according to the rules of the respective ES, whereas the integer parameters are mutated using the hybrid approach. For the mutation, a range is given, depending on the step size and the validity of the parameters, from which the new parameters are randomly drawn. Thus, in each step, all integer parameters change to a greater or lesser extent (depending on the current step size). This contrasts with the mutation of a classical GA.

Sub-study: multi-objective optimization

The domain of multi-objective optimization (MOO) is highly relevant to PSS optimization. Since, especially for PSS, several target values may be available, it is important to find algorithms that can efficiently optimize more than one fitness value at the same time. However, to be able to compare individuals in the presence of more than one fitness value, concepts are needed that allow the optimization of several target values. In this regrade, to enable individuals in a MOO to be compared with each other the concept of Pareto dominance is introduced. An individual $\boldsymbol{x}_1$is Pareto dominant $\succ$ (here, the aim is to maximize fitness) compared to another individual $\boldsymbol{x}_2$if it fulfils the following condition (Glasmachers et al., 2014; Narzisi et al., 2006):

$$\boldsymbol{x}_1 \succ \boldsymbol{x}_2 \; iff \; \left\{ \begin{array}{l} f_i(\boldsymbol{x}_1) \geq f_i(\boldsymbol{x}_2) \quad \forall \; i \; \in \{1; \ldots; k\} \\ \exists j \in \{1; \ldots; k\} f_j(\boldsymbol{x}_1) > f_j(\boldsymbol{x}_2) \end{array} \right\} \tag{1}$$

To divide the individuals of a population into ranks regarding their Pareto dominance, the condition given in Eq. (1) is checked in pairs for all individuals of the population. All individuals that are not dominated belong to the first rank of the population. All individuals which do not dominate any other individual belong to the last rank. This leads to the formation of so-called Pareto fronts. Within a Pareto front, the individuals cannot directly be compared by means of this primary sorting criterion, since no Pareto dominance exists. To still find the worst individuals within a front, the so-called *hypervolume indicator* can be used as a secondary sorting criterion. It determines the hypervolume of the fitness values, the so-called *contributed hypervolume*, which is dominated solely by the respective point. The larger the *contributed hypervolume*, the better the individual is compared to its neighbours in the same Pareto front (Loshchilov & Glasmachers, 2016; Narzisi et al., 2006; Suttorp et al., 2009). According to this metric, the selection pressure can be implemented. During this investigation, different approaches are compared while the most relevant findings are presented in the next Section.

For all investigations whenever possible, the same set of hyperparameters is chosen to perform the analysis. If this is not possible, common values for the hyperparameters are used. No hyperparameter training is performed on any of the algorithms to ensure comparability to non-tuned algorithms.

Sub-study: summary of the findings from the presented sub-studies

The experiments regarding the general suitability of the different algorithms, presented in Fig. 7, show, that the *cma.fmin* by *pycma* performs significantly worse than GD and RS. While GD rarely converges to comparably better optima than RS, RS shows a much more stable and overall advantageous performance and will therefore be used as a benchmark for further investigations. The performance of the *cma.fmin* algorithm also leads to the conclusion, that general EA need further adaptation to be well-suited for the optimize of PSS configurations.

Considering the combined findings of all experiments within the sub-studies, GA have shown to be superior to ES in many aspects when aiming to optimize PSS. For example, in the investigation regarding general suitability, the fitness progressions of GA show, that they have better properties for the treatment of local optima. Whereas the (1 + 1)-ES, in particular, is more capable of converging into the local optima of an attractor using a lower optimization budget. However, the ES cannot benefit from the CMA. The reason for this is probably the relatively low number of fitness evaluations available and the highly inhomogeneous fitness landscape of the PSS simulations. Altogether the plus selection has clearly superior properties compared to the comma selection and is particularly necessary for the Pareto optimization.

More attention should be paid to finding attractors of a good optimum than to strict convergence into an already found bad attractor. This is particularly important regarding more complex optimization tasks with even more irregular fitness landscapes. However, the algorithm should also be able to converge on promising attractors. To achieve this behaviour, population-based algorithms like the classic ($\mu + \lambda$)-GA and variations are used, as they represent the desired behaviour in the best possible way. Their fitness progressions clearly show that they change the attractors of the optima much more frequently than, for example, the (1 + 1)-ES, to achieve further fitness progress. This is particularly advantageous for complex problems. Furthermore, the ($\mu + \lambda$)-GA achieves the highest final fitness value of all algorithms and fits the other advantageous properties due to the above-mentioned characteristics.

When optimizing dependent variables, the holding method offers better properties regarding more continuous optimization progress while using population-based algorithms. Therefore, and because of the higher fitness values achieved, this method is preferred to the loop method.

The optimization of different variable types likewise shows that population-based algorithms have better properties for more complex tasks. Thus, this sub-study also argues for the use of ($\mu + \lambda$)-algorithms. ES and ES/GA compete when deciding which method to use to handle different types of variables. However, due to the significantly better properties of GA regarding population-based algorithms, the decision is made in favour of the algorithms with the larger proportion of GA. This contrasts the fact that the ES achieves higher overall fitness values with slightly greater variance.

In addition to the basic benefits of GA, they show superior suitability for the MOO (see Fig. 8). In this context, ES

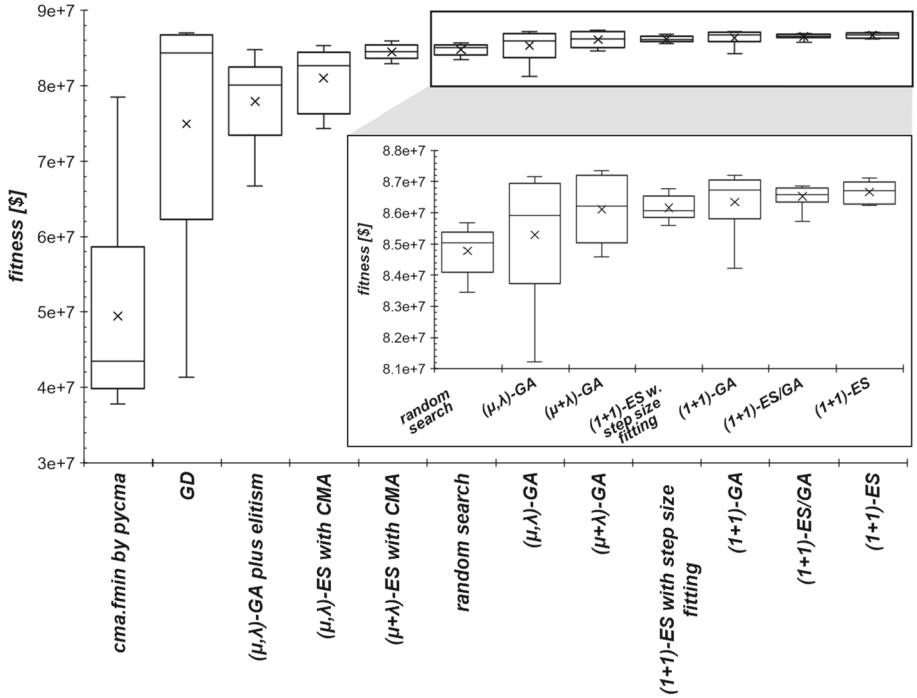

Fig. 7 Results of the sub-study regarding the general suitability of the different algorithms; each algorithm is ten times evaluated; all algorithms had the same fitness-evaluation budget; for each algorithm, the cross marks the mean, while the horizontal line within the box represents the median of each ten individual runs

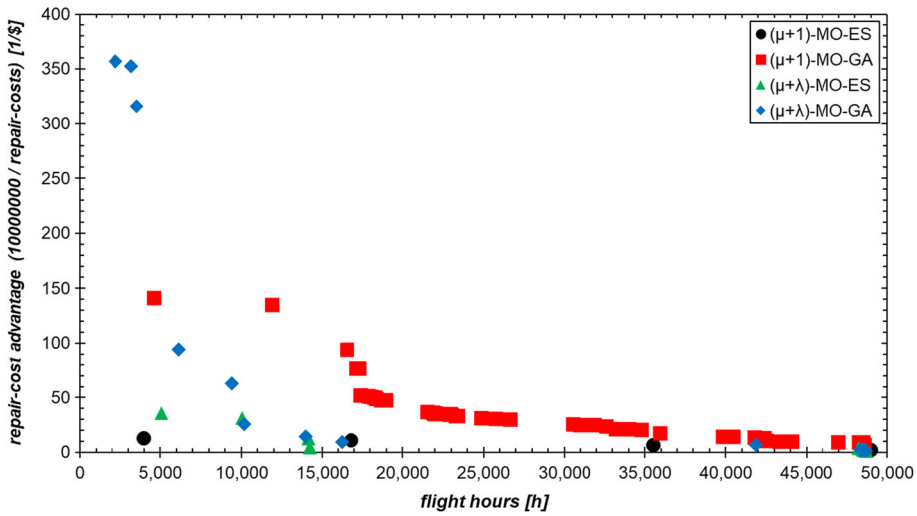

Fig. 8 Results of the sub-study regarding the Pareto optimization—showing a clear advantage in favor of GA compared to ES while both steady state ($\mu + 1$) algorithms and ($\mu + \lambda$) algorithms perform reasonably. The fitness on the vertical axis is computed based on the inverse of the repair cost to state the task as a positive maximization problem and scaled to level the importance within the hypervolume calculation

should be usually avoided for PSS optimizations, since even the use of a normalized secondary sorting criterion cannot compensate for the fundamental disadvantages of ES identified in terms of MOO. Regarding the optimization of more complex PSS simulations, a decision must be made whether there is a greater interest in better-optimized fitness values or in a broader distribution across the Pareto front. Depending on this, steady-state or conventional population-based GA should be used. For the latter, a normalization of the contributed hypervolume provides a further improvement of the MOO results.

Main-study

To exploit the findings from the sub-studies and to verify the compatibility of the individual methods, a much more complex PSS simulation is modeled. Furthermore, this study investigates whether the difference in the results between the benchmark and the final algorithm is significantly greater than in the sub-studies. And thus, whether the combination of optimized methods provides an additional benefit in PSS optimization. First, the simulation is briefly presented. Second, the combined algorithm is introduced, and built upon this the experiments and findings are presented and discussed.

Table 1 Design choices for the combined algorithm (bold are the chosen options)

Basic strategy	**GA**
	ES
Selection from the previous generation	Centre of gravity
	Parental selection
Offspring generation	**Inheritance**
	Mutation
Population size	$\mu = 1$
	$\lambda = 1$
	$\boldsymbol{\mu \geq 1}$
	$\boldsymbol{\lambda \geq 1}$
Environmental selection	**Plus**
	Comma
Dependent variables	**Holding**
	Loop
Various variable types	ES + round
	ES/GA
	ES/GA interval
MOO	**Yes**
	No
Strategy adaptation	**Yes**
	No

Main-study: simulation model

To realize this study, a PSS simulation of a car-sharing business model is set up. This simulation does not claim to be a perfect illustration of a possible real car sharing concept. It contains the elementary structures of car-sharing without being oriented toward an actual scenario. As already explained earlier, the basic suitability of EA algorithms as black-box optimization tools for PSS is to be examined in the context of this work, which is carried out on abstract PSS. Due to the invariance properties of the algorithms, it is assumed, that the findings can be transferred to real-world inspired simulative setups. Nevertheless, the car-sharing model used is geared as closely as possible to real implementations. It still offers a much larger extent and a much greater optimization difficulty than all the models used in the sub-studies.

The simulation uses a combination of all the modelling approaches for PSS presented in the Sect. "Simulation of product-service system". The basic processes of the simulation are mapped using an SD approach. This approach controls, for example, the movements of the vehicles within the model. However, since the entities used in this model are not all identical but have their own state variables that influence their behaviour and interaction with the environment, an ABM strategy is also pursued. Additionally, parts of DES can be found mainly in the modelling of vehicle relocation strategies.

The simulation offers the possibility to combine all separately investigated challenges of PSS optimization within one model and has a solution space of approximately 5.5×10^{31} without considering continuous variables. A broad representation of the overall structure of the simulation can be found in the appendix.

Main-study: combined optimization framework

Based on the findings a new algorithm—the *MO-GA-ES-MEESS* – is developed which has the following parts and properties (Table 1).

The first step to construct the algorithm is to decide which basic type of algorithm should be used. The choice is between GA and ES. Both have shown that they can achieve good results in terms of basic PSS optimization, especially in a hillclimber ($\mu = 1$; $\lambda = 1$) configuration. However, it should be noted that following the results presented in Fig. 7, GA showed to be more suitable in terms of avoiding convergence to poor local optima compared to ES when $\mu > 1$ and $\lambda > 1$. As independent of the type of environmental selection GA outperformed their respective ES counterpart. This alone justifies using GA as a basis only in part. Another crucial reason is the much better suitability of GA in MOO. Since it cannot be generally assumed that a PSS will only be optimized considering one target variable, the findings of the sub-studies are elementary for PSS optimization. The presented algorithm is capable of finding Pareto fronts within three dimensions.[2] Also, the population type is determined by general consideration and the findings of the sub-studies. Since a completely new algorithm should not be developed for every application, as the effort to design new algorithms for each use-case is very high depending on the application and knowledge of the user, the algorithm presented in this chapter should be able to fulfil different optimization tasks after a small adaptation/hyperparameter tuning to the application. Therefore, the population type is fixed to a $(\mu + \lambda)$-strategy, since additionally, it can function as a $(1 + 1)$-strategy as well as a steady-state $(\mu + 1)$ strategy. Further findings from the sub-studies concern the handling of dependent variables. As already mentioned, the holding method is used for this purpose. The main reason for this is the better optimization result of this procedure. However, this method is additionally modified. As further findings show, PSS can suffer from the fact that some configuration types are significantly less suitable for certain objectives than others. To save optimization budget, a change of the configuration type is no

[2] In a three-dimensional optimization, no secondary sorting criterion is used to avoid favouring too strong preferences towards certain target values/areas.

Table 2 Pseudocode of the developed algorithm

Algorithm 1: MO-GA-ES-MEESS

```
input:  d ≜ dimensions of the search space; μ ∈ ℕ⁺; λ ∈ ℕ⁺; X ∈ ℝ^{μ×d}; initial_mutation_rate ∈ [0; 100];
        inheritance_rate ∈ [0; 100]; initial_σ ∈ ℝ⁺; tournament_size ∈ ℕ⁺; fitness_function( );
        fitness_dimension ∈ [1; 3] ; budget ∈ ℕ⁺; reevaluation_number ∈ ℕ⁺; x_min ∈ ℝ^d; x_max ∈ ℝ^d;
        firstrank( ); lastrank( ); minimalContributor( )
additional parameter:  Y ∈ ℝ^{(μ+λ)×d}; fX ∈ ℝ^μ; fY ∈ ℝ^{(μ+λ)}; mu_rate_X ∈ [0; 100]^{μ×d} ;
                       mu_rate_Y ∈ [0; 100]^{(μ+λ)×d}; σ_X ∈ ℝ^{μ×d}; σ_Y ∈ ℝ^{(μ+λ)×d}
repeat for all j ∈ μ:                                   #initialization
  fX_j ← fitness_function(X_j, reevaluation_number)     #denoised fitness estimate
  mu_rate_Y_j, mu_rate_X_j ← initial_mutation_rate      #individual mutation rate
  σ_Y_j, σ_X_j ← initial_σ                              #individual standard deviation
repeat for all i ∈ round(budget/λ):                     #optimization step
  Y ← X                                                 #search population
  fY ← fX                                               #fitness of search population
  repeat for all n ∈ λ:
     repeat until new valid offspring is found:
        if Uniform_Distribution(0,99) ≥ inheritance_rate:   #do only mutation
           if Pareto optimization:
              Y_{μ+n} ← random x ∈ firstrank(X)
           else:
              Y_{μ+n} ← random x ∈ X                    #random selection
           σ_Y_{μ+n}, mu_rate_Y_{μ+n} ← σ_X, mu_rate_X of selected x ∈ X
           for c in range(d):                           #mutation
              if d_c = unused dependent variable: #holding
                 continue
              if Uniform_Distribution(0,99) < mu_rate_Y_{μ+n}:
                 if d_c is discrete:
                    Y_{μ+n_c} ← Uniform_Distribution(x_min_c; x_max_c)
                 else:
                    Y_{μ+n_c} ← max(x_min_c, min(x_max_c, Y_{μ+n_c} + σ_Y_{μ+n_c} × 𝒩(0, I)))
        else:                                           #parental selection
           if Pareto optimization:
              P_0 , P_1 ← firstrank_selection(firstrank(X))
           else:
              P_0 , P_1 ← tournament_selection(X)
           for c in range(d):                           #crossover inheritance
              if d_c = not used dependent variable or
              Uniform_Distribution(0,1) = 0:
                 Y_{μ+n_c} ← P_{0_c}
              else:
                 Y_{μ+n_c} ← P_{1_c}
        if Y_{μ+n} is feasible and not already contained in Y:
           fY_{μ+n} ← fitness_function(Y_{μ+n}, reevaluation_number)
           break
  if Pareto optimization:
     find the λ worst individuals ∈ Y according to Pareto dominance and
     hypervolume contribution by lastrank(Y) and/or minimalContributor(Y)
  else:
     sort Y and fY according to fY from highest to lowest and find the λ
     worst individuals ∈ Y

  exclude the λ worst individuals as environmental selection step to select
  the best offspring Y ∈ ℝ^{μ×d}
  for k in range(μ):                                    #selection
     if Y_k was chosen while parental selection: #strategy adaptation
        adapt mu_rate_Y_k and σ_Y_k based on evaluated success of the
        offspring
     X_k, fX_k, mu_rate_X_k, σ_X_k ← Y_k , fY_k, mu_rate_Y_k, σ_Y_k    #new population
```

Fig. 9 Experiment-Framework

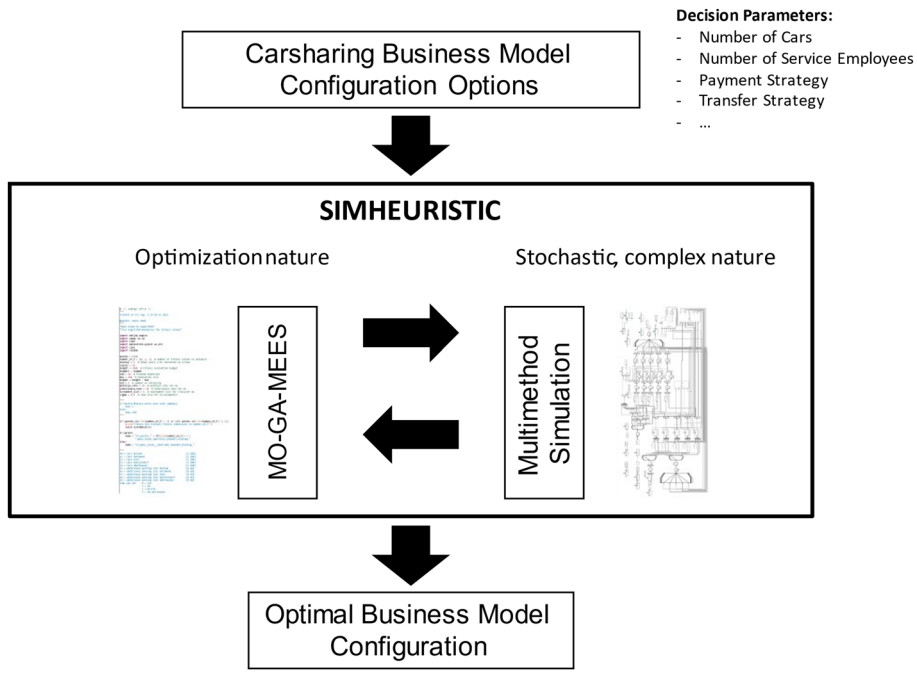

Table 3 Hyperparameters of the experiment

Dimensions	Mutation rate in %	Inheritance rate in %	Tournament size	σ	μ	λ
One	30	50	6	1	15	5
Two	30	50	–	1	100	5
Three	30	50	–	0.8	100	5

longer always made with a static chance but now depends on the mutation rate of the algorithm. This increases the chance that once the algorithm has decided on a meta configuration type, the optimization will then be more targeted. Of course, this carries the risk of getting stuck in local optima caused by the different meta configurations. Given the very high complexity and the inherently good characteristics of GA to avoid such a situation, this procedure nevertheless seems to be meaningful to avoid wasting optimization budget. Due to the previous decision for a primary GA-based optimization, it is not reasonable to use the strategy of the (μ + λ)-ES for the simultaneous optimization of different variable types. Therefore, the ES/GA-strategy is used, in which the optimization of real-valued and integer variables takes place independently of each other. However, it is significant that the GA character of the optimization is maintained by the fact that a change of the real-valued variables by the standard normal distribution is additionally dependent on the mutation and inheritance properties of the algorithm. The pseudocode for the combined approach can be found in Table 2.

Main-study: experiment

Figure 9 shows the experiment framework. The multimethod-simulation model is used to optimize decision variables like the number of cars, the number of service employees, payment strategy, transfer strategy, etc. using the MO-GA-MEESS algorithm. The algorithm's hyperparameters were set according to Table 3.

For the one-, two- and three-dimensional optimization, a budget of 1.000 fitness evaluations is granted where each fitness value is re-evaluated 10 times to deal with fitness noise. As the time needed for each fitness evaluation is several magnitudes greater than the computation time of each evolution loop, the wall clock time for the optimization of EA and random search are roughly similar. Therefore, the benchmark results are generated with the same fitness budget and re-evaluation metrics to guarantee a fair comparison. This is also aligned with most real-world optimization problems, where the fitness evaluation takes several minutes while the computation effort for the calculations of the EAs is handled in milliseconds.

The hyperparameters for the combined algorithm are not tuned to the needs of this specific scenario, as the goal is to find a general approach suitable for optimizing PSS without too much overhead. Therefore, the performance must be sufficient without extensive hyperparameter tuning for each individual PSS development.

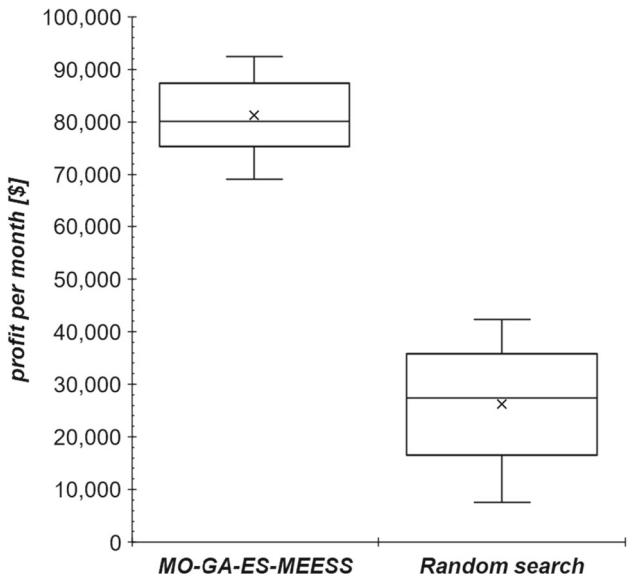

Fig. 10 Comparison of the final fitness; standard deviation for MO-GA-ES-MEESS is 6,991; standard deviation for random search is 10,842

Main-study: results

The results are compared to an RS as a benchmark, as the sub-studies showed, that PSS optimization problems with such high solution spaces cannot be solved efficiently and stable by e.g., gradient descent/ascent or general CMA-ES approaches.

The in-detail investigation of the optimization behaviour shows in general that the search points of the new algorithm are only at the start of the optimization in the same region as the majority of RS solutions. During the optimization, the search predominantly moves into areas with profit. Nevertheless, points with heavy losses are still frequently found, which shows that there are many unprofitable configurations in the direct neighbourhood of profitable configurations. This increases the difficulty of optimization as the fitness landscape is highly inhomogeneous. Further, it becomes apparent that the algorithm is quite capable of finding different attractors based on the current population, which improves its ability to escape bad local optima. Figure 10 shows the final results. To assess the optimization behaviour of the procedures on a broad basis, the results of ten individual runs are presented. The generally achieved fitness level of the EA-based optimization is significantly higher than that of the benchmark. Furthermore, the achieved variance of the results is considerably lower. Both characteristics argue for a successful optimization which is superior to the RS.

The optimization results of the two-dimensional optimization are presented in Fig. 11. The best configurations from all ten individual runs of each method are compared. It is noticeable that the MO-GA-ES-MEESS also offers clear advantages over the RS in the domain of two-dimensional optimization. Thus, the final Pareto front of the algorithm is significantly better than that of the benchmark in almost all areas of the front. Especially in those areas where the PSS is profitable, the algorithm achieves significantly higher customer satisfaction. Furthermore, the MO-GA-ES-MEESS achieves much higher profits for almost all separate configurations with matching customer satisfaction found by the benchmark, which suggests that its optimization is significantly better adapted to the conditions of the PSS. Only if customer satisfaction exceeds 230 the RS can find some more profitable configurations. However, these occur in comparatively small numbers and are of no interest within a decision process due to the extreme loss associated with it.

Within the framework of three-dimensional optimization, the results of the ten runs are summarized in one chart to ensure clarity. Since three-dimensional optimization requires a three-dimensional representation of the results, Fig. 12 shows a three-dimensional plot of the Pareto fronts achieved over the ten runs of benchmark and MO-GA-ES-MEESS. The points of the benchmark are marked with a cross, while those of the algorithm are represented by a circle. To make the three-dimensional representation more tangible to the viewer, the points are colour coded. This is based on profit and goes from dark grey (loss) to light grey (profit) for the benchmark and from dark cyan (loss) to light cyan (profit) for the algorithm. Since the third fitness value for this optimization corresponds to the number of dissatisfied customers to be minimized, the best possible configurations point is indicated by the red star.

The results shown are comparable to those of two-dimensional optimization. In configurations with high profits, the algorithm is always characterized by better customer satisfaction and fewer dissatisfied customers as well as higher profits. Here the front is much better covered and much more optimized, which can be seen from the fact that the points of the MO-GA-ES-MEESS clearly dominate the RS in all perspectives.

Discussion

In a final evaluation of the optimization behaviour and optimization results of the MO-GA-ES-MEESS (also in comparison to the benchmark), it is noticeable that the advantages of EA-based optimization are generally enhanced by the greater complexity of the PSS. Both the results of the main study and further experiments based on modified simulations show that the optimization results are significantly better than those of the benchmark. Compared to the results from the sub-study according to the general suitability of different algorithms, the difference in the optimization result between RS and EA-based optimization significantly increases for the

investigated applications. While the RS achieves comparable fitness in some runs of the sub-studies, the results of the main-study show a completely different picture. The RS does not achieve configurations as nearly as good as the algorithm. This can be determined for both single-objective and multi-objective optimization. One reason for this is the increasing complexity of the optimization task. The greater the difficulty of finding good configurations, the greater the advantages of targeted optimization. A second conducted experiment supports this assumption. Here, the probabilities of finding profitable configurations are reduced, which is proven by the performance of the RS. Basically, the algorithm continues to act very well when faced with this optimization task, which shows that it can deal with even more difficult fitness landscapes. However, increased complexity also increases the probability of convergence into poor local optima. It is, therefore, to be expected that the performance of EA-based algorithms will decline again in comparison to the benchmark

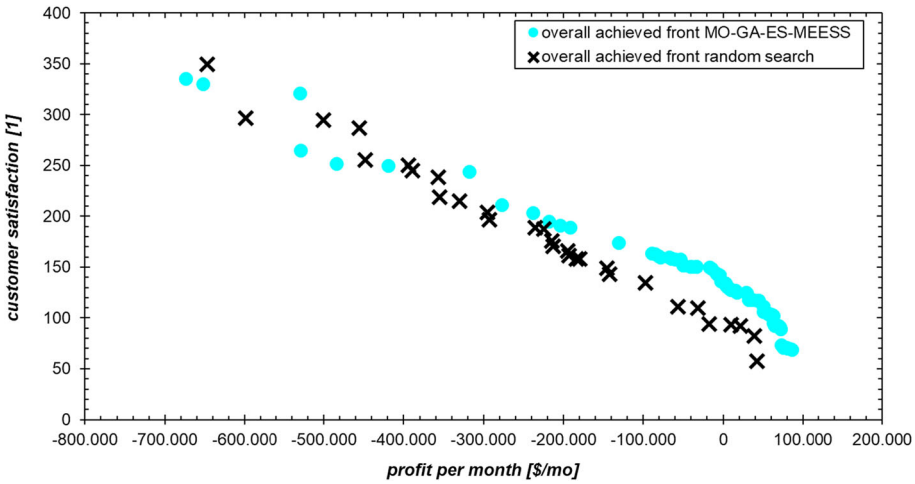

Fig. 11 Comparison of the Pareto optimization in two dimensions—The customer satisfaction is simulated based on the average waiting time of the customers, the number of unsatisfied customers, the type/quality of vehicle used within the PSS, the average aversion of the customers due to e.g., dirty cars and the current level of the pricing model compared to other providers. This function is not intended to provide a completely correct representation of customer satisfaction, but merely to pick up the trends resulting from the input variables and make them accessible for simulation and optimization

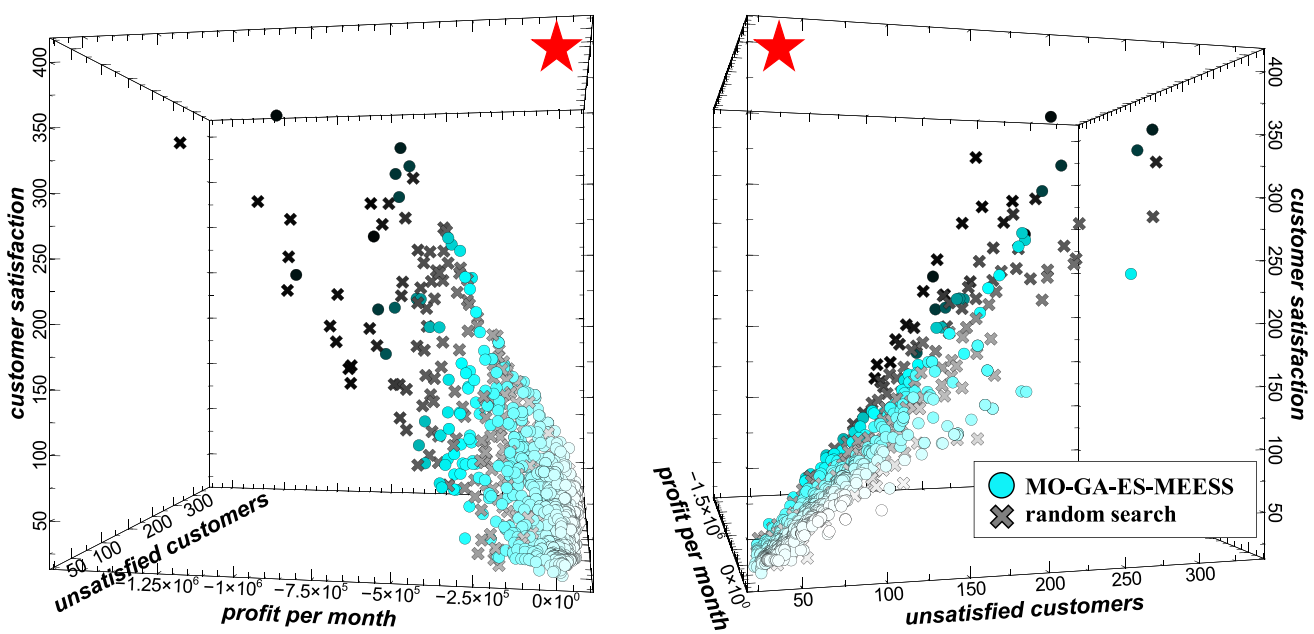

Fig. 12 Results of the three-dimensional optimization; the results of the benchmark are represented by crosses, those of the MO-GA-ES-MEESS by circles; a lower coloration implies a higher profit; the ideal direction of the optimization is marked with a red star (Color figure online)

for extremely inhomogeneous problems with small attractors that do not show any tendencies in their fitness landscape. However, this is to be expected for almost all black-box optimization problems, since no targeted optimization can take place without recognizable tendencies or sufficient attractors. In such cases, an algorithm will usually not perform better than an RS.

The reproducibility of the optimization behaviour and the optimization results are quite stable. Therefore, it can be assumed that comparably good configurations are always found with a similar probability by the algorithm. Together with the general invariance properties of EA algorithms and the additional invariance indicated by the second setup for the main experiment, the algorithm is expected to achieve similarly good results on comparable, less or even more complex tasks. This suggests a broad scope for EA-based PSS optimization, where the algorithms are clearly superior to RS optimization. Finally, the use of an EA-based approach for PSS optimization can be recommended.

Conclusions and future work

In this article, the development of a new approach for the optimization of PSS business model configurations during the design phase has been presented. For this, first, the suitability of different basic algorithms and strategies was separately tested in a simheuristic framework regarding their performance to optimize PSS business model configurations. These insights were joined to design and develop a new optimization algorithm called MO-GA-ES-MEESS. This algorithm is a multi-objective-focused genetic algorithm that aims to solve the major challenges in the design of optimization strategies for the development and improvement of PSS. Therefore, it considers the huge solution space, multi-objective optimization, heterogenous optimization parameters, and direct dependencies on design decisions and corresponding variables, which belong to the challenging nature of the PSS design phase. For validation and performance measurement purposes, the new algorithm was then applied within a simheuristic approach for the business model optimization of a car-sharing concept. The results for the single objective optimization exceeded the benchmarks by a significant margin. Also, in the context of multi-objective optimization, clear advantages measured by multiple fitness functions are achieved by the algorithm compared to the benchmark results. Our experiments show in the first place that the simheuristic approach and the new optimization algorithm are suitable for the design and development of PSS. Secondly, it gets evident, that the two-step development approach for the new algorithm was suitable to reduce the overall complexity in the algorithm design. In addition, the combination of the independent findings during the substudies proved to be possible in general. However, proof of optimality for the chosen constellation for the new algorithm concerning other possible combinations of different investigated building blocks cannot be provided and would be the subject of future work.

The simheuristic approach developed in this study offers significant advantages, primarily in two aspects. Firstly, it automates and optimizes the complex decision-making process during the design of PSS business models. Secondly, these decisions are tested and quantitatively evaluated through realistic simulation experiments that forecast the use-phase of the PSS. This forecasting capability reduces the uncertainties associated with costs and profitability, providing valuable insights for decision-makers and thus goes beyond qualitative design methodologies that end with a mentally evaluated concept of a business model. While the simheuristic approach offers numerous benefits, it does come with a few drawbacks. One such drawback is that its effective implementation often requires a high level of expertise in simulation modelling and optimization algorithms. Additionally, the calculations for assessing the fitness of each configuration can be time-consuming. However, this drawback is not critical since the design phase primarily involves strategic decisions, which are typically less time-sensitive than those made in the operational phase.

The presented approach serves as a theoretical contribution to the research need of linking data science with engineering methods for the design and optimization of PSS business models. Its generality and adaptability offer the opportunity to adapt to new PSS business models. Hence, for practitioners like PSS designers, the presented approach could be used to optimize their business model configurations and may thereby lead to better-suited configurations soon.

This article paves the way for further work. First, the application of this approach is not limited to the design and development phase. Rather, it could be used to optimize existing business models in the operational phase. Thereby, knowledge and empirical data could be used to restrict the stochastic nature of the simulation model as well as to model the customer perception more precisely. Further, the use of the simheuristic approach could also be applied to other PSS-related aspects for example service delivery planning which could lead to the generation of more robust plans. Moreover, this approach could be integrated into a bigger risk management process for PSS design and development. Insights from risk management could reduce the possible number of sensible solutions even before the application of this approach. Additional potential for the future development of the current work lies in the further elaboration of the underlying algorithms. The metaheuristic algorithm developed in this work could be evolved into a hyper-heuristic approach. By doing this, the algorithm would not search in the solution space but

in a heuristic space to find the best heuristic to generate the optimal solution, whereby the strategies presented in Section *Sub-study* could be implemented as these heuristics. Finally, the results of this optimization approach can be compared to other optimization techniques e.g., reinforcement learning.

Acknowledgements We would like to express our sincere appreciation to the reviewers for their valuable feedback and insightful comments that significantly improved the quality of this manuscript.

Author contributions First author, HM: conceived, designed the algorithms and performed the simulations, analysed the data, wrote the article. Second author, MH: design and supervision. Third author, EA: theoretical embedding in the PSS development and simheuristic domain, wrote the article. Fourth author, BK: supervision and proof reading.

Funding Open Access funding enabled and organized by Projekt DEAL. The authors received no financial support for the research, authorship, and/or publication of this article.

Data availability The data that support the findings of this article are available from the corresponding author upon request.

Declarations

Competing interests 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.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

Appendix

See Figs. 13, 14 and 15, Tables 4, 5 and 6.

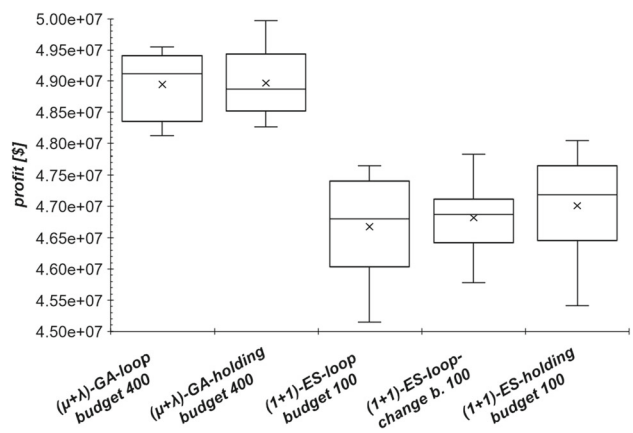

Fig. 13 Results of the sub-study regarding the optimization of dependent variables

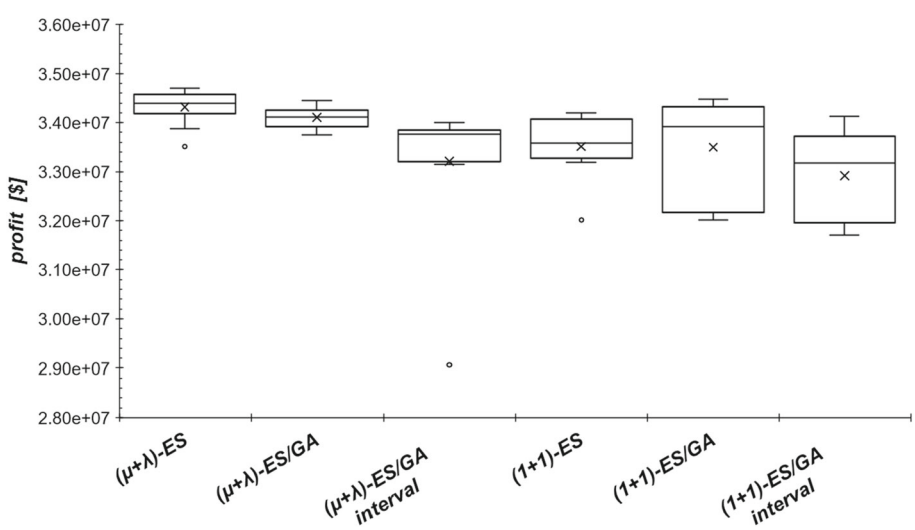

Fig. 14 Results of the sub-study regarding the optimization of various variable types

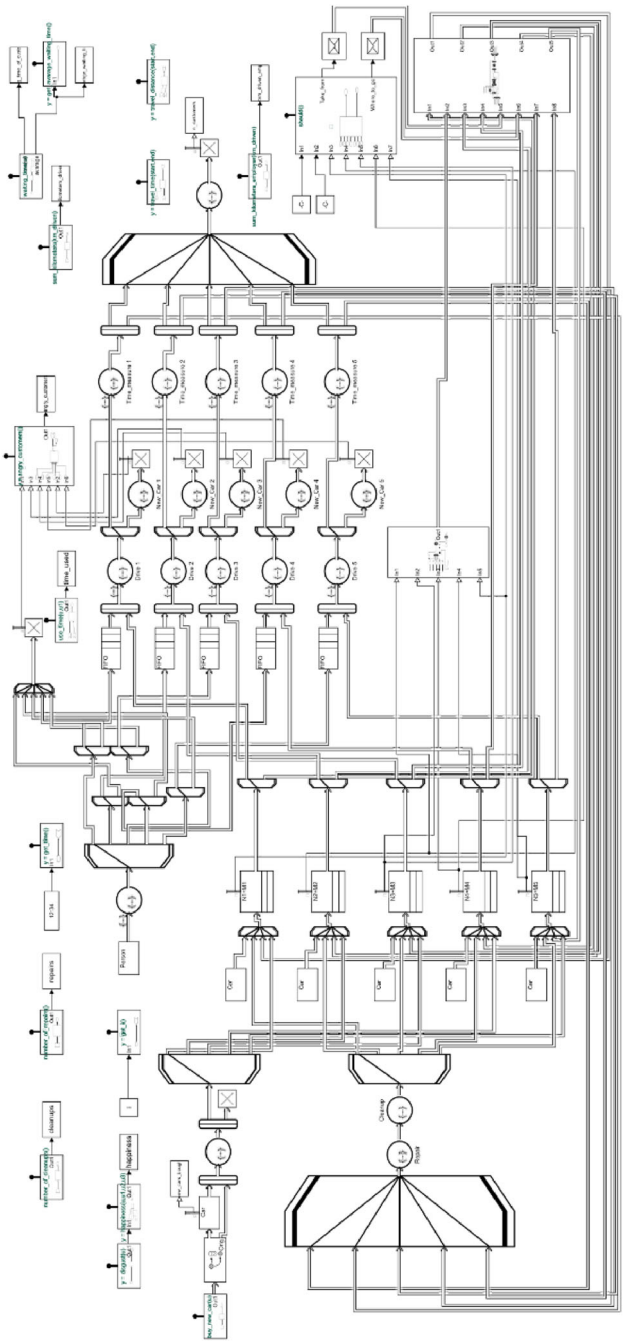

Fig. 15 Structure of the car sharing simulation based on MATLAB Simulink

Table 4 Parameter overview of the PSS simulation to determine the general suitability of different algorithms

Parameter	Explanation	Range of values
Planes	Number of aircraft initialized at the beginning	{1; 2; …; 100}
S_L	Number of service stations in London	{1; 2; …; 5}
T1_L	Number of technicians per station in London	{1; 2; …; 50}
S_M	Number of service stations in Madrid	{1; 2; …; 5}
T1_M	Number of technicians per station in Madrid	{1; 2; …; 50}
S_F	Number of service stations in Frankfurt	{1; 2; …; 5}
T1_F	Number of technicians per station in Frankfurt	{1; 2; …; 50}
S_P	Number of service stations in Paris	{1; 2; …; 5}
T1_P	Number of technicians per station in Paris	{1; 2; …; 50}

Table 5 Parameter overview of the PSS simulation for dependent optimization – PM stands for predictive maintenance

Parameter	Explanation	Range of values
overwatch	0 = not using PM 1 = using PM	{0; 1}
T1	Number of technicians in London without PM	{1; 2; …; 50}
T2	Number of technicians in Madrid without PM	{1; 2; …; 50}
T3	Number of technicians in Frankfurt without PM	{1; 2; …; 50}
T4	Number of technicians in Paris without PM	{1; 2; …; 50}
T5	Number of technicians in Rom without PM	{1; 2; …; 50}
O1	Number of technicians in London with PM	{1; 2; …; 50}
O2	Number of technicians in Madrid with PM	{1; 2; …; 50}
O3	Number of technicians in Frankfurt with PM	{1; 2; …; 50}
O4	Number of technicians in Paris with PM	{1; 2; …; 50}
O5	Number of technicians in Rom with PM	{1; 2; …; 50}

Table 6 Parameter overview of the PSS simulation to investigate strategies for different variable types; real-valued variables are *float*

Parameter	Explanation	Range of values
Planes	Number of aircraft serviced	{1; 2; …; 100}
S_L	Number of service stations in London	{1; 2; …; 5}
T1_L	Level of maintenance per station in London	(0; 50]
S_M	Number of service stations in Madrid	{1; 2; …; 5}
T1_M	Level of maintenance per station in Madrid	(0; 50]
S_F	Number of service stations in Frankfurt	{1; 2; …; 5}
T1_F	Level of maintenance per station in Frankfurt	(0; 50]
S_P	Number of service stations in Paris	{1; 2; …; 5}
T1_P	Level of maintenance per station in Paris	(0; 50]

References

Abualigah, L., Elaziz, M. A., Khasawneh, A. M., Alshinwan, M., Ibrahim, R. A., Al-qaness, M. A. A., et al. (2022). Metaheuristic optimization algorithms for solving real-world mechanical engineering design problems: A comprehensive survey, applications, comparative analysis, and results. *Neural Computing and Applications, 34*, 4081–4110. https://doi.org/10.1007/s00521-021-06747-4

Alice, R., Giuditta, P., & Cavalieri, S. (2018). Quantitative assessment of service delivery process: Application of hybrid simulation modelling. *IFAC-PapersOnLine, 51*, 1113–1118. https://doi.org/10.1016/j.ifacol.2018.08.454

Alp, E., Arnold, D., Herzog, M., & Kuhlenkötter, B. (2023). Multimethod Simulation for the Risk Management of Product-Service Systems. *Procedia CIRP, 118C*(2023), 92–97. https://doi.org/10.1016/j.procir.2023.06.01

Alp, E., Herzog, M., & Kuhlenkötter Michael. A Simulation-based Investigation of the Beneficiary of Adopting Remote Assisted Service Processes for the Providers of Industrial Product-Service Systems (IPS^2). *Proceedings of Spring Servitization Conference 2021*.

Andrew, A. M. (1998). Modern heuristic search methods. *Kybernetes, 27*, 582–585. https://doi.org/10.1108/k.1998.27.5.582.3

Bäck, T., & Schwefel, H.-P. (1993). An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation, 1*, 1–23. https://doi.org/10.1162/evco.1993.1.1.1

Bala, B. K., Arshad, F. M., & Noh, K. M. (2017). *System dynamics: Modelling and simulation*. Springer.

Beyer, H.-G., & Schwefel, H.-P. (2002). Evolution strategies—A comprehensive introduction. *Natural Computing, 1*, 3–52. https://doi.org/10.1023/A:1015059928466

Bi, H., Lu, F., Duan, S., Huang, M., Zhu, J., & Liu, M. (2020). Two-level principal–agent model for schedule risk control of IT outsourcing project based on genetic algorithm. *Engineering Applications of Artificial Intelligence, 91*, 103584. https://doi.org/10.1016/j.engappai.2020.103584

Boßlau, M. (2021). Business model engineering for smart product-service systems. *Procedia CIRP, 104*, 565–570. https://doi.org/10.1016/j.procir.2021.11.095

Brissaud, D., Sakao, T., Riel, A., & Erkoyuncu, J. A. (2022). Designing value-driven solutions: The evolution of industrial product-service systems. *CIRP Annals, 71*, 553–575. https://doi.org/10.1016/j.cirp.2022.05.006

Burke, E. K., & Kendall, G. (2014). Introduction. In E. K. Burke & G. Kendall (Eds.), *Search Methodologies.* Springer.

Caldeira, R. H., & Gnanavelbabu, A. (2021). A simheuristic approach for the flexible job shop scheduling problem with stochastic processing times. *SIMULATION, 97*, 215–236. https://doi.org/10.1177/0037549720968891

Calvet, L., Juan, A. A., Fernandez-Viagas, V., & Framinan, J. M. (2016). Combining simulation with metaheuristics in distributed scheduling problems with stochastic processing times. In *2016 Winter Simulation Conference (WSC), Washington, DC, USA, 11.12.2016 - 14.12.2016* (pp. 2347–2357). IEEE. doi:https://doi.org/10.1109/WSC.2016.7822275.

Chica, M., Angel, A. J., Bayliss, C., Cordon, O., & Kelton, W. D. (2017). Why Simheuristics? Benefits, limitations, and best practices when combining metaheuristics with simulation. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.2919208

Chih, M. (2023). Stochastic stability analysis of particle swarm optimization with pseudo random number assignment strategy. *European Journal of Operational Research, 305*, 562–593. https://doi.org/10.1016/j.ejor.2022.06.009

Coello Coello, C. A. (2002). Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering, 191*, 1245–1287. https://doi.org/10.1016/S0045-7825(01)00323-1

Dokeroglu, T., Sevinc, E., Kucukyilmaz, T., & Cosar, A. (2019). A survey on new generation metaheuristic algorithms. *Computers & Industrial Engineering, 137*, 106040. https://doi.org/10.1016/j.cie.2019.106040

Friedrich, S. (2016). Stephen Friedrich on our CorporateCare services. Edited by Rolls-Royce.

Glasmachers, T. (2013). A natural evolution strategy with asynchronous strategy updates. *GECCO 2013 - Proceedings of the 2013 Genetic and Evolutionary Computation Conference*. doi:https://doi.org/10.1145/2463372.2463424.

Glasmachers, T. (2017). Global Convergence of the (1+1) Evolution Strategy to a Critical Point. *Evolutionary Computation*. https://doi.org/10.1162/evco_a_0024

Glasmachers, T., Naujoks, B., & Rudolph, G. (2014). Start Small, Grow Big? Saving Multi-objective Function Evaluations. In T. Bartz-Beielstein, J. Branke, B. Filipič, & J. Smith (Eds.), *Parallel problem solving from nature - PPSN XIII*: *13th international conference, Ljubljana, Slovenia, September 13 - 17, 2014 ; proceedings* (Vol. 8672, pp. 579–588, Lecture Notes in Computer Science, Vol. 8672). Cham: Springer.

Glover, F., & Laguna, M. (1999). Tabu Search. In P. M. Pardalos & D.-Z. Du (Eds.), *Handbook of Combinatorial Optimization: Volume 1–3* (pp. 2093–2229). Springer, US.

Gok, Y. S., Tomasella, M., Guimarans, D., & Ozturk, C. (2020). A Simheuristic Approach for Robust Scheduling of Airport Turnaround Teams. In *2020 Winter Simulation Conference (WSC), Orlando, FL, USA, 14.12.2020 - 18.12.2020* (pp. 1336–1347). IEEE. doi:https://doi.org/10.1109/WSC48552.2020.9383947.

Hansen, N. (2006). The CMA Evolution Strategy: A Comparing Review. In J. A. Lozano, P. Larrañaga, I. Inza, & E. Bengoetxea (Eds.), *Towards a New Evolutionary Computation* (Vol. 192, pp. 75–102). Springer.

Hansen, N., Niederberger, A., Guzzella, L., & Koumoutsakos, P. (2009). A method for handling uncertainty in evolutionary optimization with an application to feedback control of combustion. *IEEE Transactions on Evolutionary Computation, 13*, 180–197. https://doi.org/10.1109/TEVC.2008.924423

Helbing, D. (2012). Agent-Based Modeling. In D. Helbing (Ed.), *Social self-organization: Agent-based simulations and experiments to study emergent social behavior* (pp. 25–70). Springer.

Juan, A. A., Faulin, J., Grasman, S. E., Rabe, M., & Figueira, G. (2015). A review of simheuristics: Extending metaheuristics to deal with stochastic combinatorial optimization problems. *Operations Research Perspectives, 2*, 62–72. https://doi.org/10.1016/j.orp.2015.03.001

Keine gen. Schulte, J., & Steven, M. (2012). Risk Management of Industrial Product-Service Systems (IPS2) – How to Consider Risk and Uncertainty over the IPS2 Lifecycle? *19th CIRP International Conference on Life Cycle Engineering,* 37–42. doi:https://doi.org/10.1007/978-3-642-29069-5_7.

Kimita, K., Tateyama, T., & Shimomura, Y. (2012). Process simulation method for product-service systems design. *Procedia CIRP, 3*, 489–494. https://doi.org/10.1016/j.procir.2012.07.084

Köster, M. (2015). *Ein Beitrag zur modellbasierten Systempartitionierung industrieller Produkt-Service Systeme*. Zugl.: Bochum, Ruhr-Univ., Diss., 2014 (1st ed., Ingenieurwissenschaften). München: Verl. Dr. Hut.

Krause, O., & Glasmachers, T. (2015). A CMA-ES with Multiplicative Covariance Matrix Updates. In A. I. Esparcia-Alczar (Ed.), *GECCO '15: Genetic and Evolutionary Computation Conference, Madrid Spain, 11 07 2015 15 07 2015* (pp. 281–288, ACM Digital Library). New York, NY: ACM. https://doi.org/10.1145/2739480.2754781.

Li, A. Q., Kumar, M., Claes, B., & Found, P. (2020). The state-of-the-art of the theory on product-service systems. *International Journal of Production Economics, 222*, 107491. https://doi.org/10.1016/j.ijpe.2019.09.012

Loshchilov, I., & Glasmachers, T. (2016). *Anytime Bi-Objective Optimization with a Hybrid Multi-Objective CMA-ES (HMO-CMA-ES)*. https://arxiv.org/pdf/1605.02720.

Medini, K., Boucher, X., Peillon, S., & Da Silva, C. M. (2015). Product service systems value chain configuration—A simulation based approach. *Procedia CIRP, 30*, 421–426. https://doi.org/10.1016/j.procir.2015.02.099

Meier, H., Roy, R., & Seliger, G. (2010). Industrial product-service systems—IPS 2. *CIRP Annals, 59*, 607–627. https://doi.org/10.1016/j.cirp.2010.05.004

Mersch, B., Glasmachers, T., Meinicke, P., & Igel, C. (2006). Evolutionary Optimization of Sequence Kernels for Detection of Bacterial Gene Starts. In Kollias S., Stafylopatis A., Duch W., & Oja E. (Eds.), *Artificial Neural Networks—ICANN 2006* (2006th ed., pp. 827–836, Lecture Notes in Computer Science, Volume 4132). Springer

Michalewicz, Z., & Schoenauer, M. (1996). Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary Computation, 4*, 1–32. https://doi.org/10.1162/evco.1996.4.1.1

Mitchell, T. M. (1997). *Machine Learning*. WCB/McGraw-Hill.

Moro, S. R., Cauchick-Miguel, P. A., Mendes, G. H., & d. S. (2021). Literature analysis on product-service systems business model: A promising research field. *Brazilian Journal of Operations & Production Management, 19*, 1–18. https://doi.org/10.14488/BJOPM.2021.043

Musa, E. O., Greasley, A., & Albores, P. (2017). The Application of Simulation to Product-Service Systems: A Review (wip). *SummerSim 2017 - Society for Modeling and Simulation (SCS) International*.

Narzisi, G., Mysore, V., & Mishra, B. (2006). Multi-objective evolutionary optimization of agent-based models: An application to emergency response planning. *Proceedings of the Second IASTED International Conference on Computational Intelligenze*, 228–232.

Nikolaus Hansen, yoshihikoueno, ARF1, Kento Nozawa, Luca Rolshoven, Matthew Chan, et al. (2022). *CMA-ES/pycma: r3.2.2*. Zenodo.

Osterwalder, A., & Pigneur, Y. (2013). *Business model generation: A handbook for visionaries, game changers, and challengers*. Wiley&Sons.

Panadero, J., Juan, A. A., Bayliss, C., & Currie, C. (2020). Maximising reward from a team of surveillance drones: A simheuristic approach to the stochastic team orienteering problem. *European J. of Industrial Engineering, 14*, 485. https://doi.org/10.1504/EJIE.2020.108581

Reim, W., Parida, V., & Örtqvist, D. (2015). Product-Service Systems (PSS) business models and tactics—A systematic literature review. *Journal of Cleaner Production, 97*, 61–75. https://doi.org/10.1016/j.jclepro.2014.07.003

Reyes-Rubiano, L. S., Faulin, J., Calvet, L., & Juan, A. A. (2017). A simheuristic approach for freight transportation in smart cities. In *2017 Winter Simulation Conference (WSC), Las Vegas, NV, 03.12.2017 - 06.12.2017* (pp. 3346–3357). IEEE. https://doi.org/10.1109/WSC.2017.8248051.

Rodrigues, S., Bauer, P., & Bosman, P. A. (2014). A novel population-based multi-objective CMA-ES and the impact of different constraint handling techniques. In C. Igel & D. V. Arnold (Eds.), *GECCO '14: Genetic and Evolutionary Computation Conference, Vancouver BC Canada, 12 07 2014 16 07 2014* (pp. 991–998). New York, NY, USA: ACM. https://doi.org/10.1145/2576768.2598329.

Salcedo-Sanz, S. (2009). A survey of repair methods used as constraint handling techniques in evolutionary algorithms. *Computer Science Review, 3*, 175–192. https://doi.org/10.1016/j.cosrev.2009.07.001

Salimans, T., Ho, J., Chen, X., Sidor, S., & Sutskever, I. (2017). Evolution Strategies as a Scalable Alternative to Reinforcement Learning.

Smith, D. J. (2013). Power-by-the-hour: The role of technology in reshaping business strategy at Rolls-Royce. *Technology Analysis & Strategic Management, 25*, 987–1007. https://doi.org/10.1080/09537325.2013.823147

Spears, W. M., Jong, K. A. de, Bäck, T., Fogel, D. B., & Garis, H. de. (1993). An overview of evolutionary computation. *Brazdil P.B. (eds) Machine Learning: ECML-93. ECML 1993. Lecture Notes in Computer Science (Lecture Notes in Artificial Intelligence), 1993*(Volume 667).

Suttorp, T., Hansen, N., & Igel, C. (2009). Efficient covariance matrix update for variable metric evolution strategies. *Machine Learning, 75*, 167–197. https://doi.org/10.1007/s10994-009-5102-1

Wang, W., Jiang, Y., Peng, Y., Zhou, Y., & Tian, Q. (2018). A simheuristic method for the reversible lanes allocation and scheduling problem at smart container terminal gate. *Journal of Advanced Transportation, 2018*, 1–14. https://doi.org/10.1155/2018/1768536

Weicker, K., & Weicker, N. (2003). Basic principles for understanding evolutionary algorithms. *Fundamenta Informaticae, 2003*(XXI), pp. 1001–1017.

Wen, H., Wang, S. X., Lu, F. Q., Feng, M., Wang, L. Z., Xiong, J. K., et al. (2022). Colony search optimization algorithm using global optimization. *The Journal of Supercomputing, 78*, 6567–6611. https://doi.org/10.1007/s11227-021-04127-2

Whitley, D. (2001). An overview of evolutionary algorithms: practical issues and common pitfalls. *Information and Software Technology, 43*, 817–831.

Woldesenbet, Y. G., Yen, G. G., & Tessema, B. G. (2009). Constraint handling in multiobjective evolutionary optimization. *IEEE Transactions on Evolutionary Computation, 13*, 514–525. https://doi.org/10.1109/TEVC.2008.2009032

Wrasse, K., Hayka, H., & Stark, R. (2015). Simulation of product-service-systems piloting with agent-based models (outlined revision). *Procedia CIRP, 30*, 108–113. https://doi.org/10.1016/j.procir.2015.02.150

Yan, K., Cheng, T. C. E., Li, G., & Wei, Z. (2021). Overcoming the service paradox by leveraging organizational design and cultural factors: A combined configuration and contingency approach. *IEEE Transactions on Engineering Management, 68*, 498–512. https://doi.org/10.1109/TEM.2019.2911105

Yan, T., Lu, F., Wang, S., Wang, L., & Bi, H. (2023). A hybrid metaheuristic algorithm for the multi-objective location-routing problem in the early post-disaster stage. *Journal of Industrial and Management Optimization, 19*, 4663–4691. https://doi.org/10.3934/jimo.2022145

Yazdani, M., Mojtahedi, M., & Loosemore, M. (2020). Enhancing evacuation response to extreme weather disasters using public transportation systems: A novel simheuristic approach. *Journal of Computational Design and Engineering, 7*, 195–210. https://doi.org/10.1093/jcde/qwaa017

Zhang, J., Zhan, Z., Lin, Y., Chen, N., Gong, Y., Zhong, J., et al. (2011). Evolutionary computation meets machine learning: A survey. *IEEE Computational Intelligence Magazine, 6*, 68–75. https://doi.org/10.1109/mci.2011.942584

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.