Cognitive Computation (2024) 16:2674–2700
https://doi.org/10.1007/s12559-024-10294-0

RESEARCH

Counterfactual Explanations in the Big Picture: An Approach for Process Prediction-Driven Job-Shop Scheduling Optimization

Nijat Mehdiyev[1,2] · **Maxim Majlatow**[1,2] · **Peter Fettke**[1,2]

Received: 16 December 2023 / Accepted: 2 May 2024 / Published online: 30 May 2024

Abstract
In this study, we propose a pioneering framework for generating multi-objective counterfactual explanations in job-shop scheduling contexts, combining predictive process monitoring with advanced mathematical optimization techniques. Using the Non-dominated Sorting Genetic Algorithm II (NSGA-II) for multi-objective optimization, our approach enhances the generation of counterfactual explanations that illuminate potential enhancements at both the operational and systemic levels. Validated with real-world data, our methodology underscores the superiority of NSGA-II in crafting pertinent and actionable counterfactual explanations, surpassing traditional methods in both efficiency and practical relevance. This work advances the domains of explainable artificial intelligence (XAI), predictive process monitoring, and combinatorial optimization, providing an effective tool for improving automated scheduling systems' clarity, and decision-making capabilities.

Keywords Explainable artificial intelligence (XAI) · Predictive process monitoring · Combinatorial optimization · Counterfactual explanations

Introduction

The use of algorithmic and data-driven solutions has rapidly increased in fields where high-stakes decisions are made [1]. This growing dependence highlights the importance of transparency and interpretability in artificial intelligence (AI) systems [2]. A notable development in explainable AI (XAI) is the adoption of counterfactual explanations. This method explains AI decisions by proposing minimal adjustments to input features that could have changed the outcome. Counterfactual explanations have been successfully used in different application domains, such as finance, healthcare diagnostics, customer service, and risk assessment [3, 4]. While these explanations excel in handling queries individually, their effectiveness diminishes in complex, system-level decision-making contexts [5, 6]. Resource allocation in healthcare, urban planning, production scheduling, supply chain management, and construction scheduling are examples of domains where decision-making extends beyond individual instances. Decisions made in such scenarios can have far-reaching effects that span across a network of interconnected entities, influencing the entire system. Therefore, an expanded view of counterfactual explanations is needed that considers not only individual elements within the system but also the overarching "big picture". For instance, optimizing patient care in healthcare could strain resources, or focusing on single-operation efficiency in production scheduling might lead to less effective overall sequences. Recognizing this, our study introduces a novel framework for counterfactual explanations integrating systemic considerations. This approach broadens the scope from individual instances to encompass system-wide complexities, aiming to ensure that enhancements in parts contribute positively to the entire system's efficiency.

More specifically, we explore the generation of counterfactual explanations within combinatorial optimization-based decision-making, specifically focusing on improving the Job-Shop Scheduling Problem (JSSP). We employ predictive process mining coupled with a multi-objective opti-

✉ Nijat Mehdiyev
nijat.mehdiyev@dfki.de

Maxim Majlatow
maxim.majlatow@dfki.de

Peter Fettke
peter.fettke@dfki.de

[1] German Research Center for Artificial Intelligence (DFKI), Campus D 3.2, Saarbrücken 66123, Saarland, Germany

[2] Saarland University, Campus D 3.2, Saarbrücken 66123, Saarland, Germany

mization method for counterfactual searching. This approach begins with analyzing historical process execution data obtained from a Manufacturing Execution System (MES), from which we develop a machine learning (ML)-based process prediction model. This model estimates the processing times for operations in the planned jobs. Following this, we create an initial plan utilizing the Mixed Integer Linear Programming (MILP) algorithm. The next phase involves conducting a counterfactual search at the individual operation level of the examined schedule, employing a multi-objective optimization strategy. The Non-dominated Sorting Genetic Algorithm (NSGA-II) employed not only aims to enhance the processing times of individual operations by altering feature values but also strives to optimize the total makespan while maintaining adherence to the predetermined schedule. In every step of this novel counterfactual search, MILP for JSSP is invoked to compute the makespan, effectively integrating these findings into the ongoing optimization loop for the counterfactual search. This comprehensive approach aims to refine individual operation efficiencies while concurrently enhancing the global efficiency of the entire production plan.

In the given manufacturing scenario, let's say the initial plan, developed using MILP and informed by predictions from the predictive process monitoring model, assigns Operation A to Machine X and Worker Y, with an anticipated processing time of 2 h. However, through the predictive model, it is determined that if Operation A had been assigned to Machine Z and Worker W, the processing time could have been reduced to 1.5 h. The counterfactual explanation, generated using the NSGA-II algorithm, might then propose: *The initial plan assigned Operation A to Machine X and Worker Y, resulting in a processing duration of 2 h and a total makespan of 10 h for the schedule. If Operation A had been assigned to Machine Z and Worker W, with the processing duration predicted to be 1.5 h, and all other operations and assignments remained constant, the total makespan of the schedule would have been reduced to 9 h and 45 min.* This explanation demonstrates how altering the combination of workers and machines for specific operations can lead to improved processing times. It provides a practical scenario where a slight adjustment in the resource allocation (in this case, the worker and machine assigned to an operation) can lead to significant gains in overall efficiency while still maintaining the balance and coherence of the entire manufacturing schedule. Therefore, in this scenario, changes that only enhance the processing duration of an operation without improving the overall schedule's makespan would not be considered for counterfactual explanations.

In our research, we adopt granular computing as a structured thinking and information processing framework, pivotal for our counterfactual identification process [7]. Granular computing enables the expression of both input and output at varying levels of granularity, offering a multi-layered perspective essential for nuanced decision-making. This approach allows us to methodically dissect the complexities of the JSSP into more manageable segments. By categorizing the scheduling process into different levels of granularity — from individual operations (representing the highest level of granularity) to broader job categories, and ultimately to the entire scheduling system (the most aggregated view) — we achieve a detailed yet holistic problem formulation (see Fig. 1). This tiered perspective is instrumental in identifying the requirements for counterfactuals at each level, ensuring that our optimization strategies are not only relevant and precise at the micro-level but also harmonious and effective at the macro-scale. Such a structured approach to granular computing empowers us to precisely tailor counterfactual suggestions, enhancing the overall efficacy and adaptability of manufacturing schedules in response to dynamic operational conditions.

The remainder of this paper is structured as follows: "Background and Related Work" section offers an overview of foundational principles and recent advancements in XAI, focusing on counterfactual explanations and their significance in system-level analysis and JSSP, with an emphasis on the application of granular computing. The "Methodology" section outlines the methodology used, followed by the experimental setup in the "Experiment Settings" section. The "Results" section details the outcomes of the machine learning, optimization, and counterfactual explanation components of the proposed approach, with a particular focus on showcasing and evaluating the multi-objective identification of counterfactual explanations. The "Discussion" section discusses the practical and scientific implications, while the "Conclusion" section presents concluding remarks.

Background and Related Work

XAI has emerged as a crucial research domain, addressing the growing need for transparency and understanding in AI systems [8]. The essence of XAI lies in its ability to make the decision-making processes of complex AI models comprehensible to humans [1]. This necessity stems from various factors, including ethical considerations, regulatory compliance, and the practical need to trust and effectively manage AI solutions [9]. A significant area within XAI is the generation of counterfactual explanations [4]. These explanations present alternative scenarios that would lead to different outcomes, helping users understand the model's decision boundaries. The definition of counterfactual explanations aligns closely with human reasoning, as they answer "what-if" questions that are intuitively understood [10].

The recent studies on counterfactual explanations highlight several key desiderata such as validity, sparsity, plausibility, actionability, and diversity [4, 11]. Validity requires

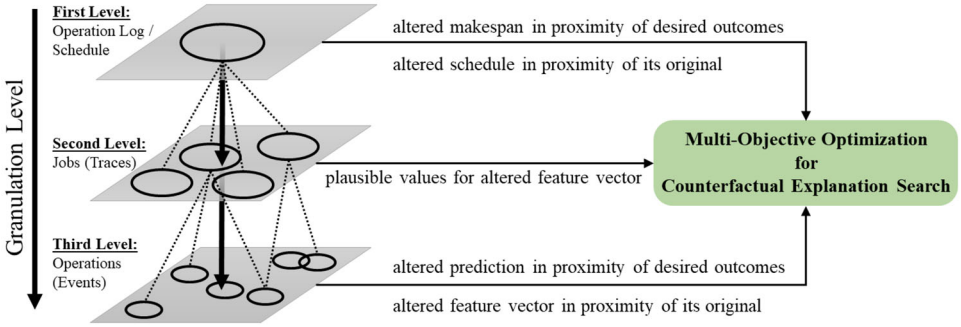

Fig. 1 Overview of the proposed uncertainty explainability approach

counterfactuals to match the model's decision-making. Sparsity emphasizes minimal changes from the original instance, simplifying and interpreting the counterfactual. Plausibility ensures counterfactuals are realistic and believable. Actionability makes counterfactual suggestions practical and implementable by focusing on the user's ability to act. Finally, diversity promotes multiple counterfactuals to clarify the model's decision boundaries. Achieving one of these desiderata can compromise another, so they must be balanced. For instance, sparsity may reduce explanation plausibility or diversity. In general, counterfactual-based methods prioritize soundness, whereas data-summarizing techniques favor completeness despite being less sound [12]. Our study addresses these trade-offs by creating holistic counterfactual explanations that meet these critical requirements.

The literature on the evaluation counterfactual explanations includes various methods and measures for quality assessment. Emphasis is placed on the importance of practicality and robustness in evaluation processes [13]. Additionally, the significance of robust and plausible explanations for enhancing individual fairness is highlighted [14]. The necessity for creating diverse and actionable explanations is also underscored, along with the introduction of frameworks for their generation and metrics for assessment [12]. The literature further advocates for user testing and qualitative analysis in evaluations, identifying key deficits in existing methodologies [15]. Overall, there is a call for a more robust and comprehensive approach to the evaluation of counterfactual explanations within XAI.

Identifying counterfactual explanations is essentially a search or optimization task, aiming to find an alternative close to the original data point and plausible in real-world scenarios [16]. The optimization method is based on creating a loss function that includes the desired properties and then using well-known optimization algorithms to lower this function [4]. The authors of [3] conducted a groundbreaking study in this area, where they were among the early proponents of a framework for counterfactual explanations. This approach is further expanded upon in subsequent studies. In [17], authors presented a method called distribution-aware counterfactual explanation, which is distinguished by its utilization of a loss function that integrates the Mahalanobis distance and the local outlier factor to evaluate the credibility of potential counterfactuals. In a similar vein, [18] introduced a model-agnostic approach designed to handle various tabular data and different distance functions. The authors of [12] made a noteworthy contribution with their approach which imposes constraints that guarantee both feasibility and diversity in the generated counterfactuals. The recent work of [19] introduced an interactive system that offers visual counterfactual explanations. Their approach optimizes a loss function by taking into account various factors such as validity, minimal distance, and diversity of changes.

Search-based methods identify counterfactuals by minimizing a cost function in each iteration. A model-agnostic heuristic approach for textual data that uses best-first search with pruning for local improvements was proposed in [20]. In [21], the authors added a SHAP-based heuristic for counterfactual generation and [22] introduced a Multi-Objective Counterfactual explanation (MOC) [22]. As a multi-objective genetic optimization problem, MOC generates diverse counterfactuals that balance objectives. A visual counterfactual explainer that uses simple heuristics for minimal changes to provide visual counterfactual explanations for ML models was introduced in [23]. Finally, [24] proposed an approach based on a genetic algorithm that generates diverse counterfactuals with constraints for plausibility and accountability. The relevance of these explanations lies in their ability to provide actionable insights, enabling users to understand not just how the model arrived at a decision, but also how they might alter inputs to achieve a desired outcome.

An excellent overview and analysis of over 50 distinct methods in optimization and search-based approaches, which are fundamental in advancing counterfactual explainers, have been provided in [4]. Most counterfactual explanations, as outlined previously, have been investigated primarily in the context of predictive analytics, showing relevance and rigor for the cases examined. However, these explanations often do not fully meet the needs, particularly in scenarios where the systemic implications of predictions on various entities are substantial. Recognizing this gap, recent research has begun recently to focus on leveraging counterfactual explanations

as a tool for decision support in combinatorial optimization problems [5, 6]. This shift marks a significant expansion in the application of counterfactual explanations, moving beyond predictive analytics to address more complex and systemic challenges. In this context, [5] uses counterfactual explanations, focusing on minimal alterations to the reality that would yield different outcomes as suggested by the inquirer. Their method extends counterfactual explanation identification for decisions based on optimization, applying a refined version of inverse combinatorial optimization. In their study, [6] developed a mathematical programming-based method for generating counterfactual explanations in Workforce Scheduling and Routing Problem (WSRP) systems, focusing on minimal data alterations to meet user queries. This approach emphasizes efficient changes to the WSRP instance data to align outputs with user preferences.

In a similar vein, our research introduces an innovative approach in the field of counterfactual explanations, particularly focusing on system-level analysis. To our knowledge, this is the first study to generate system-level counterfactual explanations for the JSSP, a complex NP-hard problem. Distinctively, our methodology diverges from previous methods in two significant ways. Firstly, we employ a machine learning model trained on real-world data to predict a critical decision variable: the processing duration of operations. This variable is then utilized as an input for the mathematical optimization problem. Our unique "predict-then-optimize" strategy effectively combines predictive process monitoring with schedule optimization. Secondly, we propose a multi-objective optimization approach for identifying counterfactual explanations that extend beyond individual operation-level predictions, encompassing the broader schedule level. This dual-level analysis offers a more comprehensive understanding of the scheduling process, highlighting both operational and scheduling insights. Our approach thus represents a significant advancement in the application of counterfactual explanations to complex scheduling problems.

In our research, we have adhered to specific principles for identifying objectives in the search for counterfactual explanations, integrating the concept of granular computing. Granular computing recognized as an effective framework for structured thinking, problem-solving, and information processing, has seen wide application in various real-world scenarios [7, 25]. Its utility spans across fields such as data mining, fuzzy control, and security [26]. Building upon this foundation, [27] expands the application of granular computing to big data processing. The proposed multi-granularity joint problem-solving approach demonstrates its efficacy in handling large-scale data. Granular computing, a key concept in mathematical optimization, involves the allocation of information granularity to enhance model-system concept [28]. In this regard, [29] explores the modeling of rough granular computing for developing data models and addressing complex optimization problems. The utility of granular computing extends to the domain of scheduling algorithms as well. For instance, [30] implemented a granularity-based workflow scheduling algorithm tailored for cloud computing environments. This approach not only optimized the scheduling process but also enhanced performance metrics like makespan and virtual machine utilization. By incorporating granular computing into our study, we can take a more detailed approach to identifying counterfactual search objectives at various levels of granularity. Moreover, it facilitates comprehension of the broader implications of the generated explanations, ensuring a thorough and comprehensive analysis of the scheduling problems in consideration. The decision to use this methodology greatly enhances our study, providing a more profound understanding of the complexities of counterfactual explanations in scheduling settings.

Methodology

In our proposed approach, we initially train a predictive process monitoring model on historical process execution data to accurately predict operation processing durations. With these predictions, we proceed to generate an initial schedule using MILP (see Fig. 2). The subsequent phase involves generating counterfactual explanations through the application of the NSGA-II for multi-objective optimization. In each iteration of this process, the algorithm selects an individual operation, assessing it against multiple objectives: conventional counterfactual goals like proximity and reduction in processing duration, along with the overall makespan and sequence similarity to the original schedule. Notably, each iteration includes optimizing the schedule to determine the obtained makespan. Modifications are made to the feature vector of the chosen operation while maintaining the status of other operations, thus incorporating these variables into an integrated multi-objective counterfactual search. This strategy aims not only to enhance the efficiency of individual operations but also to preserve the overall effectiveness and coherence of the entire manufacturing process.

The "Predicting Processing Time of Operations" section introduces and presents the preliminaries of process mining concepts for the JSSP context and discusses the predictive modeling of processing times for operations. The "JSSP: Objective, Variables, and Constraints" section presents the mathematical optimization approach for JSSP, detailing its parameters, decision variables, objective function, and constraints. In the "Multi-objective Counterfactual Explanations" section, the focus is on multi-objective optimization, specifically identifying system-level counterfactual explanations. Finally, the "Multi-objective Counterfactual Explanations" section examines the use of the NSGA-II algorithm in addressing the multi-objective aspects of the problem.

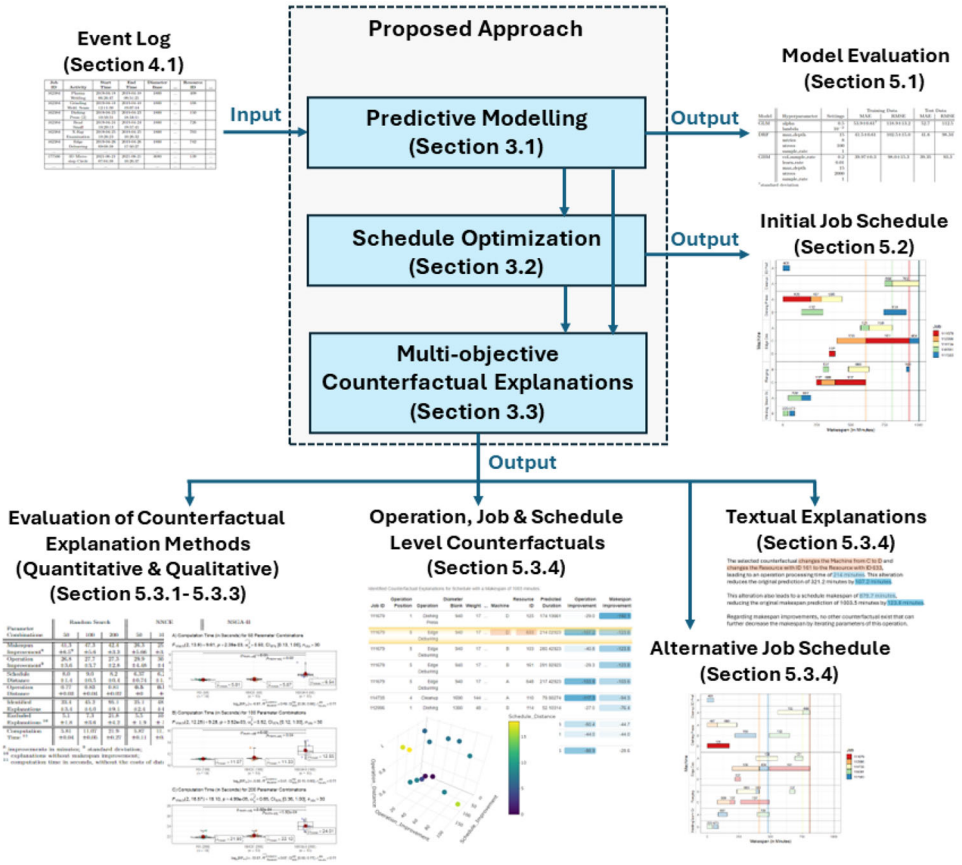

Fig. 2 Overview of the proposed multi-objective optimization approach for counterfactual explanations

Predicting Processing Time of Operations

The predictive modeling phase of our proposed approach seeks to leverage historical process event data to forecast the processing times for each operation, an aspect often overlooked in optimization strategies. This subsection introduces tailored notations and definitions derived from the domain of process mining, specifically adapted to align with the JSSP context. This customization is essential for a detailed appreciation of the complexities and dynamics prevalent in JSSP, as viewed through the process mining lens. Within this adjusted framework, "events" correspond to individual operations, "traces" are associated with jobs, and "event logs" are reinterpreted to represent operation logs. This refined approach enhances our understanding and analysis of JSSP, bridging the gap between theoretical optimization and practical application challenges. Following the formal definition of the prediction problem, we proceed to outline the ML algorithms selected to address the described regression challenge.

Definition 1 (**Operation**) An *operation* is a tuple $o = (a, j, m, r, t_{start}, t_{complete}, v_1, \ldots, v_n)$, where:

- $a \in \mathcal{A}$ represents the activity type of the operation;
- $j \in \mathcal{J}$ is the job identifier;
- $m \in \mathcal{M}$ denotes the machine on which the operation is performed;
- $r \in \mathcal{R}$ denotes the resources utilized during the operation;
- $t_{start} \in \mathbb{N}$ is the start timestamp of the operation (defined as seconds since $1/1/1970$, a Unix epoch time representation);
- $t_{complete} \in \mathbb{N}$ is the completion timestamp of the operation;
- $v_1, \ldots, v_n$ represents the list of operation-specific attributes, where $\forall 1 \leq \tilde{n} \leq n : v_{\tilde{n}} \in \mathcal{V}_{\tilde{n}}$, with $\mathcal{V}_{\tilde{n}}$ denoting the domain of the $\tilde{n}^{th}$ attribute.

Consequently, $\mathcal{O} = \mathcal{A} \times \mathcal{J} \times \mathcal{M} \times \mathcal{R} \times \mathcal{T}_{start} \times \mathcal{T}_{complete} \times \mathcal{V}_1 \times \cdots \times \mathcal{V}_n$ is defined as the universe of operations. Moreover, we define the following projection functions given the operation $o \in \mathcal{O}$:

- $p_a : \mathcal{O} \to \mathcal{A},\ p_a(o) = a,$
- $p_j : \mathcal{O} \to \mathcal{J},\ p_j(o) = j,$
- $p_m : \mathcal{O} \to \mathcal{M},\ p_m(o) = m,$
- $p_r : \mathcal{O} \to \mathcal{R},\ p_r(o) = r,$
- $p_{t_{start}} : \mathcal{O} \to \mathcal{T}_{start},\ p_{t_{start}}(o) = t_{start},$
- $p_{t_{complete}} : \mathcal{O} \to \mathcal{T}_{complete},\ p_{t_{complete}}(o) = t_{complete},$

- $p_{v_i} : \mathcal{O} \rightarrow \mathcal{V}_i, \ p_{v_i}(o) = v_i, \forall 1 \leq i \leq n$

Definition 2 (**Job**) Let $\sigma \in \mathcal{O}^*$ denote a *job*, where $\sigma_j = \langle o_1, o_2, \ldots, o_{|\sigma_j|} \rangle$ is a finite sequence of operations corresponding to a specific job j, and $\mathcal{J}$ denote the universe of jobs. Each operation o_i in σ is unique (occurs no more than once). Furthermore, for any two operations $o_i, o_{i'} \in \sigma$, the following conditions hold:

- $p_j(o_i) = p_j(o_{i'})$: This ensures all operations within σ belong to the same job, maintaining job consistency within the sequence.
- $p_{t_{start}}(o_i) \leq p_{t_{start}}(o_{i'})$ for $1 \leq i < i' \leq |\sigma_j|$: This enforces a chronological order based on the operation start timestamps, reflecting the real-world progression of tasks within the job.
- $p_{t_{complete}}(o_i) \leq p_{t_{start}}(o_{i'})$ for $1 \leq i < i' \leq |\sigma_j|$: This enforces the termination of an operation before the start of the subsequent operation pertaining to the same job.
- The total number of operations in σ_j, also known as the trace length, is represented by $|\sigma_j|$.

Definition 3 (**Partial Job Sequences**) In the context of predictive process monitoring, two methods for constructing partial sequences from a job sequence σ_j are defined as follows: *prefix* and *suffix* extraction, represented by functions $hd^i(\sigma_j)$ and $tl^i(\sigma_j)$ respectively:

- **Prefix Extraction** (hd^i): Given an integer i within the range $[1, |\sigma_j|]$, $hd^i(\sigma_j)$ constructs a prefix as the sequence $\langle o_1, o_2, \ldots, o_{\min(i,|\sigma_j|)} \rangle$, comprising the first i operations of σ_j.
- **Suffix Extraction** (tl^i): For the same integer i, $tl^i(\sigma_j)$ yields a suffix as $\langle o_w, o_{w+1}, \ldots, o_{|\sigma_j|} \rangle$ where $w = \max(|\sigma_j| - i + 1, 1)$, capturing the last i operations of σ_j.

Definition 4 (**Operation Log**) The *operation log* $\mathcal{O}_\mathcal{J}$ is defined as a set of completed jobs, $\mathcal{O}_\mathcal{J} = \{\sigma_j \mid j \in \mathcal{J}\}$.

Definition 5 (**Prediction of Processing Times**) Given a feature extraction function $\phi : \mathcal{O} \rightarrow \mathcal{X}$ that extracts job and operation-level features for each operation, thus mapping the raw data into a structured feature vector x, we define the output variable y. This variable quantifies the processing time for an operation o, calculated as $y = p_{t_{\text{complete}}}(o) - p_{t_{\text{start}}}(o)$. With this groundwork, the input feature space $\mathcal{X}$, which consists of vectors derived from operation attributes, and the output space $\mathcal{Y}$, indicative of processing times, are established. The dataset $\{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$ comprises N pairs of these vectors and their corresponding processing times. The objective is to develop a function $f : \mathcal{X} \rightarrow \mathcal{Y}$, which predicts the processing time $\hat{y}$ for a new feature vector $\hat{x}$ corresponding to an operation $\hat{o}$. The function f aims to approximate the actual processing time as precisely as possible.

To tackle the regression problem in predictive process monitoring, our study employs Gradient Boosting Machines (GBM) [31]. GBM is highly suitable for tabular data due to its ability to intricately model complex, non-linear relationships within structured datasets. This method iteratively improves prediction accuracy by building a series of decision trees, each correcting the errors of its predecessor. GBM's flexibility for parameter tuning and robustness against common data issues like missing values make it superior in handling diverse and challenging datasets. Overall, GBM's precision and adaptability render it an optimal choice for the nuanced demands of predictive process monitoring.

For benchmarking of the model performance, we examine Generalized Linear Models (GLM) [32] and Distributed Random Forests (DRF) [33], with implementations provided by the *h2o*-library (see the "Dataset Overview and Tools" section) alongside GBM. GLM stands as a traditional statistical approach, extending linear regression to accommodate various relationships through different link functions and distributions. GLM's inherent interpretability and its capacity to elucidate variable significance provide invaluable insights into process dynamics. DRF, in contrast, offers a robust ensemble method by combining multiple decision trees to enhance predictive accuracy and mitigate over-fitting. Although this model lacks inherent interpretability, the incorporation of randomness in data sampling and feature selection broadens the model's applicability, aiding in capturing a wide array of patterns, with its scalability allowing for large-scale datasets, a common scenario in process monitoring endeavors.

JSSP: Objective, Variables, and Constraints

The second stage of our proposed methodology focuses on the application of optimization algorithms to the specified scheduling challenge, guided by the forecasts provided by the predictive model. In this study, we consider a scheduling environment consisting of a finite set of jobs $\mathcal{J}$, machines $\mathcal{M}$, and workers $\mathcal{R}$. Each job $j \in \mathcal{J}$ is composed of a sequence of operations σ_j, which must be processed in a pre-determined order, respecting the precedence constraint. Operations are processed by machines and workers, where the respective activity of each operation o_{ij} is eligible to be processed by a subset of machines $\mathcal{M}_{i,j}$ and requires a worker capable of operating the corresponding machine from $\mathcal{R}_{\mathcal{M}_{i,j}}$ for its execution. The availability of jobs and machines is assumed from the start of the scheduling horizon.

The processing time for each operation, as executed by a worker on a chosen machine, is determined through an ML model. Constraints ensure that each machine and worker is

engaged in at most one operation at any time. Moreover, an operation can be executed no more than once and must wait for its preceding operations to be completed before it commences. The scheduling objective is to minimize the overall completion time, known as the makespan. This goal is pursued by solving the problem regarding the sequencing of operations. The formal definition of all other elements is described as follows:

Indices

- i: Index for operations, $i = 1, 2, \ldots, |\sigma_j|$.
- j: Index for jobs, $j = 1, 2, \ldots, |\mathcal{J}|$.
- k: Index for machines, $k = 1, 2, \ldots, |\mathcal{M}|$.
- l: Index for workers, $l = 1, 2, \ldots, |\mathcal{R}|$.

The outlined indices provide a structured approach to categorize relevant elements pertaining to the JSSP. Building upon this structure, we now turn our attention to the definition of the sets:

Sets

- $\mathcal{J}$: Set of all jobs, $\mathcal{J} = \{1, 2, \ldots, u\}$.
- $\mathcal{M}$: Set of all machines, $\mathcal{M} = \{1, 2, \ldots, v\}$.
- $\mathcal{R}$: Set of all workers, $\mathcal{R} = \{1, 2, \ldots, w\}$.
- σ_j: Set of operations for job $j, \sigma_j = \{o_{j1}, o_{j2}, \ldots, o_{j_{|\sigma_j|}}\}$.
- $\mathcal{M}_{i,j}$: Subset of machines capable of processing the activity $p_a(o_{ij})$ of operation o_{ij} of job j.
- $\mathcal{R}_m$: Subset of workers capable of operating machine m.

These sets encapsulate the aggregate elements of the scheduling environment, thereby furnishing a comprehensive view of the jobs, machines, and workers involved. Next, we progress to the definition of a central parameter:

Parameters

- $\hat{y}_{i,j,m,r}$: Processing time for operation $o_{i,j}$ using machine m and worker r, as predicted by the employed predictive model.

Decision Variables

- $p_{t_{start}}(o_{ij})$: Start time of operation o_{ij}.
- $X_{i,j,m,r}$: Binary variable that equals 1 if operation o_{ij} is assigned to machine k and worker r, 0 otherwise.

These decision variables form the operative layer of our approach, representing actionable elements within the optimization problem, and enabling manipulation and optimization of the operation scheduling. Leveraging the previous definitions, we introduce an objective function as a driving force toward optimal solutions:

Objective Function

Minimize the makespan and the completion time of the last operation of the last job.

$$\min C_{\max} \tag{1}$$

$$C_{\max} = \max_{j \in \mathcal{J}, o_{ij} \in \sigma_j} \left\{ p_{t_{start}}(o_{ij}) + \sum_{m \in \mathcal{M}_{i,j}} \sum_{r \in \mathcal{R}_k} \left(\hat{y}_{i,j,m,r} \cdot X_{i,j,m,r} \right) \right\} \tag{2}$$

This function encapsulates the primary goal, minimizing the schedule makespan, and serves as the criterion for evaluating the effectiveness of identified scheduling solutions. While the objective function provides a target for optimization, the following constraints define the space of feasible solutions:

Constraints

Event Precedence Constraint This constraint rigorously ensures that the chronological order of operations within a job is meticulously adhered to. In particular, each operation must not start before the previous operation in the job sequence is completed.

$$p_{t_{start}}(o_{ij}) + \sum_{m \in \mathcal{M}_{i,j}} \sum_{r \in \mathcal{R}_m} \left(\hat{y}_{i,j,m,r} \cdot X_{i,j,m,r} \right) \leq p_{t_{start}}(o_{(i+1)j}), \ \forall o_{ij} \in \sigma_j \setminus \{o_{|\sigma_j|j}\}, j \in \mathcal{J} \tag{3}$$

Machine and Worker Assignment Constraint Adhering to the underlying structures of the manufacturing process, each operation must be processed by exactly one machine and one worker.

$$\sum_{m \in \mathcal{M}_{i,j}} \sum_{r \in \mathcal{R}_m} X_{i,j,m,r} = 1 \quad , \forall o_{ij} \in \sigma_j, j \in \mathcal{J} \tag{4}$$

Machine and Worker Utilization Constraint Due to the nature of the underlying manufacturing scenario, a machine must be operated and monitored by a corresponding worker for the total duration of an operation. Thus, each machine and worker can only be allocated to one operation at a time.

$$p_{t_{start}}(o_{ij}) + \sum_{m \in \mathcal{M}_{i,j}} \sum_{r \in \mathcal{R}_m} \left(\hat{y}_{i,j,m,r} \cdot X_{i,j,m,r} \right) \leq p_{t_{start}}(o_{i'j'}), \quad \forall i \neq i', j \neq j', m \in \mathcal{M}, r \in \mathcal{R} \tag{5}$$

with the additional condition that if $X_{i,j,m,r} = X_{i',j',m,r} = 1$, then the operations $o_{ij}, o_{i'j'}$ cannot overlap in time.

Non-Negativity of Start Time Constraint Precluding any unrealistic scheduling scenarios, each operation must have a non-negative start time.

$$p_{t_{start}}(o_{ij}) \geq 0 \quad \forall o_{ij} \in \sigma_j, j \in \mathcal{J} \tag{6}$$

In addressing the described JSSP, this study adopts a MILP approach. The relevance of the MILP method in solving JSSP lies in its proficiency in managing complex constraints and multiple objectives. MILP excels in articulating the discrete aspects of scheduling tasks, encompassing vital constraints like sequential task ordering and resource allocation. Although alternative optimization methodologies exist, MILP distinguishes itself through its capability to deliver precise and optimal solutions within a reasonable computational timeframe, particularly in highly complex scenarios. This makes it a potent tool for addressing the complexities inherent in JSSP.

Multi-objective Counterfactual Explanations

Upon completing the training of the ML-based predictive process monitoring model and establishing an initial schedule, we advance to the third stage of our proposed approach, which entails the generation of multi-objective counterfactual explanations. The formal definition and specifics of this process are outlined below.

Definition 6 (**Counterfactual Search**) Given a predictive model $f : \mathcal{X} \rightarrow \mathcal{Y}$, with $\mathcal{Y} \in \mathbb{R}$, a set of desired operation outcomes $\mathcal{Y}' \subseteq \mathbb{R}$, a schedule $s \subseteq \mathcal{O}_{\mathcal{J}}$, and the corresponding makespan $C_{max,s} \in \mathbb{R}$, and a set of desired schedule outcomes $C'_{max,s} \subseteq \mathbb{R}$, a counterfactual explanation in form of an alternative schedule $s_{x'}$ for the original schedule s based on an altered feature vector x' for the original feature vector x of an operation o from a job $\sigma \in s$ satisfies the following conditions:

1. the prediction $f(x')$ for the altered feature vector x' lies in the proximity of desired outcomes $\mathcal{Y}'$
2. the altered feature vector x' lies in the proximity of the original feature vector x
3. the altered feature vector x' consists of plausible feature values for the underlying scenario
4. the alternative makespan $C_{max,s_{x'}}$ lies in the proximity of the of desired schedule outcomes $C'_{max,s}$ after the optimization of operation sequencing
5. the alternative schedule $s_{x'}$ lies in the proximity of the original schedule s after the optimization of operation sequencing

The *counterfactual search* is defined as a multi-objective minimization task as follows:

$$\min_{x,s_x} Obj(x, s_x) = \min_{x,s_x} \Big(Obj_1(f(x'), \mathcal{Y}'), Obj_2(x', x), Obj_3(x', \mathcal{X}^{Obs}), Obj_4(C_{max,s_{x'}}, C'_{max,s}), Obj_5(s_{x'}, s) \Big) \tag{7}$$

with $Obj : \mathcal{X} \rightarrow \mathbb{R}^5$ and $\mathcal{X}^{Obs}$ being the observed feature vector space. Obj_1 quantifies the distance between prediction $f(x')$ for the an altered feature vector x' and the desired outcomes $\mathcal{Y}'$ as follows:

$$Obj_1\left(f(x'), \mathcal{Y}'\right) = \begin{cases} 0 & \text{if } f(x') \in \mathcal{Y}' \\ \inf_{y' \in \mathcal{Y}'} |f(x') - y'| & \text{else} \end{cases}$$

Obj_2 quantifies the distance between an altered feature vector x' and its original x by employing the Gower-Distance as follows:

$$Obj_2\left(x', x\right) = \frac{1}{N} \sum_{i=1}^{N} \delta_G\left(x'_i, x_i\right) \in [0, 1] \tag{8}$$

with δ_G applying the respective measuring method to each variable type in the following manner:

$$\delta_G\left(x'_i, x_i\right) = \begin{cases} \frac{1}{\mathcal{X}_i} |x'_i - x_i| & \text{if } x'_i, x_i \text{ are numerical variables} \\ \mathbb{I}_{x'_i \neq x_i} & \text{if } x'_i, x_i \text{ are categorical variables} \end{cases} \tag{9}$$

with $\mathcal{X}_i$ being the range of applicable values for feature i, observed in $\mathcal{X}^{Obs}$. Obj_3 quantifies whether the altered feature vector x' consists of plausible values given the observed feature vectors from $\mathcal{X}^{Obs}$ as follows:

$$Obj_3\left(x', X^{Obs}\right) = \begin{cases} 0 & \text{if } x' \in X^{Obs} \\ +\infty & \text{else} \end{cases} \tag{10}$$

Obj_4 quantifies the distance between an alternative makespan $C_{max,s_{x'}}$ and its desired schedule outcomes $C'_{max,s}$ as follows:

$$Obj_4\left(C_{max,s_{x'}}, C'_{max,s}\right) = \begin{cases} C_{max,s_{x'}} - C'_{max,s} & \text{if } C_{max,s_{x'}} \in C'_{max,s} \\ +\infty & \text{else} \end{cases} \tag{11}$$

Obj_5 quantifies the distance between an altered schedule s_x and its original s by employing the Levenshtein-Distance with regards to the sequencing order of operations within the schedule as follows:

$$Obj_5\left(s_{x'}, s\right) = C_{|s_{x'}|,|s|}, \tag{12}$$

with $C_{|s_{x'}|,|s|}$ denoting the following modified version of the generic recursion for the Levenshtein distance:

$$C_{i,i'} = \min \begin{cases} 0, & \text{if } i = i' = 0 \\ C_{i-2,i'-2} + 1, & \text{if } i, i' > 0 \end{cases} \quad (13)$$

This modified version of the generic recursion is based in the condition that alternative schedules consist of the same amount of operations and only differ in the sequencing thereof. Thus, any alternative schedule is a result of a limited amount of operation transpositions from the original schedule and the difference between two sequences can be measured by the minimum amount of operation transpositions.

We employ the NSGA-II algorithm to tackle our multi-objective optimization problem, renowned for its efficacy in such scenarios due to its low computational demand, incorporation of elitism, and its capacity to sustain a diverse set of solutions [34, 35]. NSGA-II excels in optimizing multiple, often conflicting, objectives concurrently, yielding a spectrum of Pareto optimal solutions that represent the most efficient trade-offs. The process begins with the initialization of a random population, which is then organized into Pareto fronts based on fitness. The foremost front comprises solutions that are unrivaled, and deemed the "best" currently available. Selection for reproduction is based on both the non-domination rank and crowding distance of solutions, ensuring a balance between optimality and diversity. Through crossover and mutation, alongside merging with the parent population, NSGA-II incorporates elitism and ensures diversity, preparing for the next generation by reassessing dominance relationships due to new entrants. This iterative process, guided by crowding distance, prevents convergence to a singular solution, instead exploring a broad objective space. Termination is typically determined by a set number of generations. NSGA-II's strategy of balancing exploration and exploitation in the solution space ensures a comprehensive search for optimal outcomes in multi-objective contexts. Essential inputs for NSGA-II include the number of decision variables, fitness function, objective dimensions, population size, and the probabilities for crossover and mutation.

For the underlying scenario, the decision variables consist of the index of the operation that is being manipulated, an appropriate alternative machine for the selected operation, and an appropriate resource able to operate the selected machine. This information is encoded to numerical values on which genetic operations can be performed via the NSGA-II implementation. A corresponding decoder yields the operation index, machine, and resource, allowing for the analysis of non-dominated solutions. The fitness function encompasses the decoding of the encoded operation index, machine, and resource, the estimation of the processing time of the altered operation via the underlying ML model, the rescheduling of the altered operation processing time, and the subsequent evaluation of the altered schedule. Lastly, the scores for four determined objectives are returned: the improvement in processing time for the altered operation, the improvement in schedule makespan, the Gower distance between the vectors of the original and altered operation, and the Levenshtein distance between the original and altered schedule.

Experiment Settings

This section gives a general overview of the underlying dataset, the tools employed for implementing our proposed approach, hyperparameter optimization for the ML model, and a description of utilized model evaluation metrics. Moreover, it outlines the framework within which the study's experiments were conducted, detailing the dataset, implementation tools, predictive modeling, optimization approach, counterfactual explanation generation, and the metrics used for evaluating the performance of these components. This comprehensive overview ensures clarity on the methodology applied in the research, encompassing data preparation, analytical tools selection, hyperparameter tuning processes for the ML model, and the criteria for assessing the effectiveness and accuracy of the proposed counterfactual generation approach.

Dataset Overview and Tools

Table 1 illustrates a selected portion of the operational log data, providing insights into specific aspects of the operations. Table 2 presents a general statistic for the underlying operations log, which is an extended version of the dataset utilized in [36, 37]. In particular, the data was extended via an expansion of the examined time period, while also restricting its volume by filtering out article groups represented in less than 0.2% of operations.

The log consists of 161,109 operations grouped into 31,628 jobs, with each job describing the manufacturing process for a specific product. Each operation pertains to a specific manufacturing activity among a total of 15 activities. Based on product specifications, the choice and sequencing of the manufacturing steps are determined by a domain expert. Hence, each row of the dataset contains relevant information about the ordered product as well as the corresponding manufacturing process in the form of operation- and job-level information distributed across, respectively, 9 and 13 variables. The information ranges from article dimensions, weight, the processed material as well as the equipment, personnel qualified to operate the utilized machines, and the processing time for the execution of the underlying production step. Each operation is associated with

Table 1 Production operation log

Job ID	Activity	Start Time	End Time	Diameter Base	...	Resource ID	...
162384	Plasma	2019-04-18	2019-04-18	1800	...	409	
	Welding	06:26:47	09:51:25				
162384	Grinding	2019-04-18	2019-04-18	1800	...	108	
	Weld. Seam	12:11:30	19:07:14				
162384	Dishing	2019-04-23	2019-04-23	1800	...	150	
	Press (2)	10:50:31	18:34:11				
162384	Bead	2019-04-24	2019-04-24	1800	...	726	
	Small	10:20:13	19:57:45				
162384	X-Ray	2019-04-25	2019-04-25	1800	...	703	
	Examination	10:26:23	10:26:32				
162384	Edge	2019-04-26	2019-04-26	1800	...	742	
	Deburring	09:08:38	17:50:27				
...	...	...	...	...	...	...	...
177566	3D Micro-	2021-06-21	2021-06-21	3680	...	139	...
	Step circle	07:04:38	10:26:37		...		...
...	...	...	...	...	...	...	...

exactly one activity and machine that is necessary to execute the manufacturing step, as well as a resource that is qualified to operate the corresponding machine. The dataset exhibits a mean job length of 5.1 operations and a standard deviation of 3.93 operations, with a mean processing time of 109.1 min per operation and a standard deviation of 164.1 min.

For hyperparameter tuning and model training, the dataset was split into training and test sets with a ratio of 4:1, resulting in the training data consisting of 128,956 operations and the test data consisting of 32,154 operations. Details for the relevant datasets can be found in Table 2.

The implementation of the presented approach, including data processing, hyperparameter optimization, and model training, was carried out in *R*. Predominantly, the *tidyverse* library-collection was used for data cleaning and preparation in conjunction with the *h2o* library for hyperparameter tuning and model training via the *h2o Flow* interface. The implementation of the genetic algorithm was implemented utilizing the *nsga2R* library, with a customizable objective function enabling the measurement of computation time via the *tictoc* library, and the scheduling algorithm was integrated by utilizing functions from the *lpSolveAPI* library. For visualizations, *formattable*, *ggplot2*, *ggstatsplot*, and *ggbeeswarm* were used. Computations were performed on a 64-bit system using a 13th Gen Intel(R) Core(TM) i7-13700F processor with 2,100 Mhz and 16 Cores, as well as 32 GB of physical memory (random-access memory, RAM).

Table 2 Dataset information

	Training	Test	Total
Number of operations	128,956	32,150	161,106
Number of jobs	25,295	6,323	31,618
Mean trace length ± SD	5.1 ± 3.9	5.09 ± 4.6	5.1 ± 3.93
Mean processing time ± SD (in Minutes)	108.4 ± 163.0	112.0 ± 168.9	109.1 ± 164.1
Unique activities	15	15	15
Activities with alternative machines	7	7	7
Max. number of alternative machines	4	4	4
Max. number of alt. Machine operators	74	58	75

Machine Learning Settings

For predictive production scheduling scenarios, the predictive quality of a model is pivotal for the generation of alternative schedules as well as counterfactual explanations. In order to find an appropriate model and model-specific hyperparameter settings that are satisfying for the underlying scenario, a hyperparameter optimization was conducted for Generalized Linear Model (GLM), Distributed Random Forest (DRF), and Gradient Boosting Machine (GBM) algorithms. In particular, a random search was performed for each model, employing a random search of all applicable hyperparameter value combinations, with criteria for early stopping being a maximum of 20 fitted models or improvement below 0.1 min for the MSE metric after 5 consecutive rounds. The hyperparameter optimization used the training dataset for fitting, leveraging 10-fold cross-validation to mitigate overfitting and reveal imbalances in the training dataset. For the final evaluation, the models were scored on the test dataset. For each of the above models, Table 3 depicts the search spaces for each hyperparameter. For the final model, the best-performing model type with the most efficacious hyperparameter configurations was adopted.

Evaluation Metrics

In order to evaluate the model performance of the adopted machine learning approach, we scrutinize two aspects of the predictive strength. Firstly, concerning point predictions, the assessment of the predictive quality for regression models can rely on a range of metrics that endeavor to capture the prediction error in relation to the underlying ground truth value. In this study, we employ RMSE and Mean Absolute Error (MAE) as the chosen metrics. The RMSE is derived by computing the squared differences between the predicted and actual values for the given dataset, followed by calculating the square root of the average:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y_i}\right)^2} \tag{14}$$

As posited by [38], the RMSE is considerably influenced by the distribution of error magnitudes, which may render it misleading for assessing model quality in specific instances. Consequently, the authors propose MAE as a more comprehensible metric. The MAE is calculated by determining the average of the absolute differences between the predicted and actual values for the given dataset:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y_i}| \tag{15}$$

Following the suggestion in [39], where the RMSE was demonstrated to be a more reliable evaluation metric in cases with Gaussian error distribution, both metrics will be employed in this study to capture model performance.

Table 3 Hyperparameter optimization settings for GLM, DRF, and GBM models

Model	Hyperparameter	Search space
GLM	alpha	[0, 0.25, 0.5, 0.75, 1]
	lambda	$[10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10]$
DRF	max_depth[a]	[5, 10, 15]
	mtries[b]	[2, 4, 8]
	ntrees[c]	[100, 1000, 2000]
	sample_rate[d]	[0.8, 1]
GBM	col_sample_rate[e]	[0.2, 0.5, 1]
	learn_rate[f]	[0.01, 0.1]
	max_depth	[5, 10, 15]
	ntrees	[100, 1000, 2000]
	sample_rate	[0.8, 1]

[a]maximum tree depth
[b]number of selected columns at each level
[c]number of trees
[d]row sample rate
[e]column sample rate
[f] learn rate

Scheduling Scenario Settings

A scheduling scenario is constructed, providing a default schedule as a starting point and a baseline for further analysis. For the construction of the scenario, operations from jobs pertaining to the production of five different articles are sampled and the processing times of each operation are predicted by the fitted model. Afterward, the schedule is optimized using the adopted MILP approach, poses as a starting point for finding counterfactual explanations, and is leveraged to demonstrate the proposed counterfactual search approach.

In particular, the following five jobs are processed: Job 111679 produces ten pieces of the article "Torispherical Head DIN 28011" with activities performed in succession on the following machine types *Dishing Press*, *Flanging*, *Edge Deburring*, *Flanging*, *Edge Deburring*. Job 112996 produces six "Torispherical Head DIN 28011" articles but under different material specifications, using *Dishing Press*, *Flanging*, and *Edge Deburring* machines. Job 114735 produces four "Ellipsoidal Head DIN 28013" articles, utilizing the machine for *Dishing Press*, *Flanging* and *Edge Deburring*, and *Cleanup*. Job 116591 produces one "Dished Bottom," using the machine for *Welding*, *Seam Grinding*, *Dishing Press*, *Flanging* and *Edge Deburring* and *Cleanup*. Lastly, job 117583 produces one "Normal Bottom," employing *3D-Profiling*, *Welding*, *Seam Grinding*, *Dishing Press*, *Flanging*,

and *Edge Deburring*. For *Dishing Press* operations can be executed on up to four machines, *Edge Deburring* provides three machines and *Flanging* offers two machine options. The allocation of machines and workers as it was found in the underlying operations log was used as the default configuration, forwarding the construction of a viable schedule to the MILP component of the proposed approach.

Counterfactual Explanations Generation and Evaluation

Counterfactual explanations offer insight by suggesting the smallest alterations to input variables that would have led to a different prediction by the employed model, thus furnishing a direct and intuitive understanding of model behavior. Various methodologies for generating counterfactual explanations exist [4], each characterized by its unique approach to navigating the challenges inherent in explaining complex models. Techniques such as random search, nearest neighbor counterfactual explanations [10] (NNCE), and NSGA-II exemplify the diversity of strategies employed to construct meaningful and actionable counterfactuals. These methodologies differ significantly in terms of their underlying principles and the trade-offs they present between explanation simplicity, computational efficiency, and realism of the proposed counterfactuals. As such, the selection of a counterfactual generation technique must be informed by the specific context of application, taking into account the nuances of the model being explained and the objectives of the involved stakeholders.

For the evaluation of generated counterfactual explanations, the previously described scheduling scenario is leveraged as a baseline for the fitness function of the NSGA-II approach (see the "Multi-objective Counterfactual Explanations" section) and corresponding counterfactual explanations. In particular, a comparative analysis is performed against a random search and a NNCE approach, examining the quality and quantity of the generated explanations under the aspects of improvements in schedule makespan, operation processing time, distance to the original operation, and schedule, number of identified counterfactual explanations as well as the computation times.

NSGA-II NSGA-II is designed for optimizing multiple conflicting objectives simultaneously. It distinguishes itself through an efficient sorting mechanism that categorizes solutions into different fronts based on their dominance relationships, prioritizing solutions that are non-dominated and allowing for a high degree of diversity. In our implementation, NSGA-II commences with an initial population of potential counterfactuals generated through informed random mutations, adhering to operational constraints. These constraints include the suitability of machines for specific operations and the capability of resources to handle these machines. The algorithm then iteratively evolves this population over ten generations, applying selection, crossover, and mutation of genetic operators, guided by the objectives delineated in Definition 6. In particular, the crossover probability was set at 80% with a crossover distribution index set to 5, while mutation probability was set to 20% with a mutation distribution index set to 10. Through successive generations, NSGA-II refines the population, ultimately converging toward a set of optimal counterfactual schedules that satisfy the predetermined objectives.

Random Search Random search offers a straightforward and intuitive approach by randomly generating potential counterfactuals and evaluating their viability in subsequent steps, emphasizing simplicity and ease of implementation. However, its effectiveness is contingent on the search space size and may require extensive computational resources for complex models. The random search was selected as a baseline approach toward generating counterfactual explanations and implemented as follows: Operations from the default schedule are selected randomly and their manipulatable features are iterated. As with the NSGA-II implementation, these manipulations were restricted to machines viable for the operation type as well as resources being able to operate said machines. The employed ML model then scored the altered operations, estimating their processing times. In the next step, for each altered operation a corresponding alternative schedule was optimized and evaluated with regard to the objectives determined in Definition 6.

NNCE NNCE identifies counterfactuals by incorporating a distance measure during the generation of counterfactuals, thus locating the closest instances that would result in a different decision. This method benefits from its reliance on actual data points, ensuring realistic and achievable counterfactuals. Nonetheless, its efficacy is limited by the representativeness of the dataset as well as the restrictions imposed by the utilized proximity measurement. Thus, finding counterfactual explanations for the optimization across multiple objectives depends on the concrete implementation of NNCE. NNCE was selected as an advanced approach toward generating counterfactual explanations and implemented similar to random search with the following specifics: Due to the manipulatable features, in particular machine and resource, both being categorical, we employed the Gower distance as a proximity measurement, allowing any alternative feature value to be picked with equal probability. This proximity measurement yields greater distances with an increasing number of altered variable values. To ensure only the nearest neighbors are selected, we restricted variable value itera-

tions to only one of those variables. After optimizing and evaluating the alternative schedules, the nearest neighbor counterfactuals are ranked based on the defined objectives (see Definition 6).

We further examined various levels of search depth based on the number of explored parameter combinations and reiterated our tests ten times to assess the robustness of the performed approaches. In order to capture significant results, pairwise Welch's *t*-test [40, 41] are performed as well as effect size estimations via Hedges [42] and a Bayesian analysis [43] across various search depths with regards to improvements in schedule makespan and operation processing time as well as the number of identified counterfactual explanations. Finally, we evaluate and compare the properties of the generated counterfactuals in terms of validity, sparsity, plausibility, actionability, proximity, and diversity [4, 11].

Results

This section presents the results yielded by the proposed methodology and described approaches. First, the results of the hyperparameter optimization are presented, including a comparative analysis of the model performance based on the introduced metrics. Second, an exemplary scheduling scenario is illustrated as a basis for the ensuing evaluation and detailed comparison of the proposed approaches for generating counterfactual explanations. The section concludes with a comparative analysis of the random search, NNCE, and the NSGA-II approach for finding applicable counterfactual explanations.

Hyperparameter Optimization and Model Evaluation

Each of the GLM, DRF, and GBM models underwent hyperparameter optimization as described in the "Machine Learning Settings" section. The optimized settings, as well as the MAE and RMSE metrics across 10-fold cross-validation via training data and a final evaluation via test data, are documented in Table 4. The GLM, with the alpha-value of 0.5 and lambda-value of 10^{-2}, demonstrated a moderate performance with an MAE of 53.9±0.61 and an RMSE of 118.9 ± 13.2 during 10-fold cross-validation, and an MAE of 52.7 and RMSE of 112.5 on the test dataset. The DRF model, configured with a maximum tree depth of 15 (max_depth = 15), 8 selected columns at each level (mtries = 8), 100 trees (ntrees = 100), and a row sample rate of 1 (sample_rate = 1), outperformed the GLM model with a remarkably lower MAE of 41.5 ± 0.61 and RMSE of 102.5 ± 15.0 during cross-validation, with similar results on the test data (MAE = 41.6, RMSE = 98.34). The GBM model fitted with a column sample rate of 0.2 (col_sample_rate = 0.2), a learning rate of 0.01 (learn_rate = 0.01), a maximum tree depth of 15 (max_depth = 15), with 2000 trees (ntrees = 2000), and row sample rate of 1 (sample_rate = 1), surpassed other models on evaluations with training and test data. In particular, the GBM achieved the lowest MAE (39.97 ± 0.3) and RMSE (98.0 ± 15.3) during cross-validation as well as the final evaluation with the test data (MAE = 39.35, RMSE 93.3). In the context of the evaluation via test data, the GBM outperformed the GLM by a noteworthy margin, a reduction of ∼ 25% for MAE and ∼ 16% for RMSE, the differences between the GBM and DRF models are comparatively minor, exhibiting differences slightly above 5% for both MAE and RMSE. With the fitted GBM models outperforming any other model type with regard to predictive strength, the final model was trained with the identified hyperparameter settings. Corresponding hyperparameter settings and evaluation metrics are documented in Table 4.

Default Schedule with MILP

To evaluate the proposed approach, an exemplary scenario was constructed by sampling five jobs pertaining to the production of five different articles. An optimized schedule was calculated by utilizing the predictions of the adopted ML model for the estimated processing times of each involved operation (see the "Scheduling Scenario Settings" section).

Figure 3 is a Gantt chart depicting the default schedule for the production process of five different vessel bases and showcasing the operational timeline for the production of these metal components. The chart is organized by lanes, each corresponding to a specific machine, with operations scheduled along a timeline, measuring the makespan of the schedule. The operations are denoted by horizontal bars within each lane, and their lengths represent the processing time of the respective operations. The colors of these bars correspond to distinct jobs, identified by unique numerical codes, with each comprising a sequence of operations necessary to complete the manufacturing of the respective component. Vertical lines accentuate the completion time of jobs and are color-coded accordingly. Each of the 24 operations is accompanied by a label above it, depicting the identifier of the responsible worker, with 21 unique workers being assigned. The operations are distributed across eleven machines pertaining to seven machine types, *3D-Profiling*, *Cleaning*, *Dishing Press*, *Edge Deburring*, *Flanging*, *Seam Grinding* and *Welding*, offering a selection of machines for *Dishing Press*, *Edge Deburring*, *Flanging* with up to three options. The depicted jobs exhibit a diverse range of start, processing, and end times, spanning over a total of 1003 min (schedule makespan), elucidating the complexity of the scheduling environment, where various jobs are interlaced through shared resources and sequential dependencies.

Table 4 Results of the hyperparameter optimization for GLM, DRF, and GBM models

Model	Hyperparameter	Settings	Training data MAE	RMSE	Test data MAE	RMSE
GLM	alpha	0.5	53.9±0.61[a]	118.9±13.2	52.7	112.5
	lambda	10^{-2}				
DRF	max_depth	15	41.5±0.61	102.5±15.0	41.6	98.34
	mtries	8				
	ntrees	100				
	sample_rate	1				
GBM	col_sample_rate	0.2	39.97±0.3	98.0±15.3	39.35	93.3
	learn_rate	0.01				
	max_depth	15				
	ntrees	2000				
	sample_rate	1				

[a]standard deviation

Counterfactual Explanations

The analysis of the results pertaining to the generated counterfactual explanations in the given scenario is divided into two distinct research questions:

RQ1. How does the proposed multi-objective search for counterfactual explanations compare against other approaches in producing relevant counterfactual explanations?

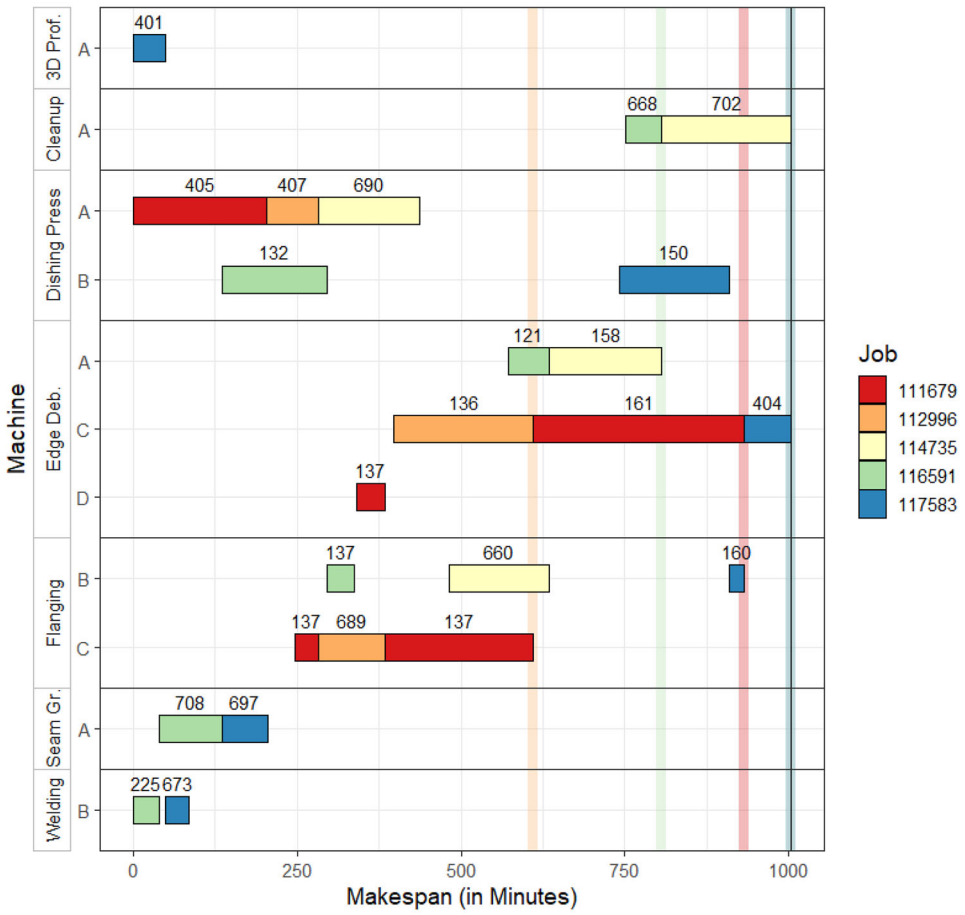

Fig. 3 Gantt chart depicting the default production schedule for the exemplary scenario

RQ2. How does the proposed approach contribute to a comprehensive exploration of the generated counterfactual explanations?

To address these questions, first, the JSSP scenario outlined in experiment settings is constructed with a default schedule as a baseline for further analysis. Next, various possible explanations for the scheduling process are generated using random search, NNCE, and NSGA-II, and compared afterward. Lastly, we examine solutions regarding schedule makespan in detail by exploring the corresponding alternative production schedules yielded by two different counterfactual explanations of the proposed approach.

Comparative Evaluation Regarding Objectives

To answer the **RQ1**, counterfactual explanations were generated by applying the proposed multi-objective NSGA-II approach, and a comparative analysis was performed against a random search and NNCE strategy. In particular, each methodology explored the alteration of the machine and resource parameters of one operation at a time and was tested across varying search depths, exploring 50, 100, and 200 parameter combinations. For the random search and NNCE, the total space of feasible parameter combinations was explored accordingly, whereas the NSGA-II approach created a population of 5, 10, or 20 parameter combinations with each of the 10 produced generations, keeping the top performers at the first Pareto front. In order to evaluate the robustness of the applied approaches, the described process was repeated ten times and results were aggregated and analyzed accordingly.

Table 5 depicts the averaged results and corresponding standard deviations with regards to the improvements of schedule makespan, operation processing time, the distances between the original and altered schedules or operations, the total number of identified counterfactual explanations, and the number of explanations without makespan improvements among the identified counterfactual explanations of each iteration. For the evaluation of the computational cost, the execution times for each approach were measured as well and are examined in the "Comparative Evaluation Regarding Computation Times" section.

Makespan Improvement NNCE exhibits the lowest average makespan improvements, ranging from 25.8 to 26.3 min, followed by random search with values ranging from 41.3 to 47.3 min. The performance does not improve with more parameter combinations, indicating a potential inefficiency in exploring the solution space. Standard deviations decrease with increasing number of explored parameter combinations, suggesting an increasing stability of random search and NNCE with increasing search depth. Considering the nature of the NNCE approach, the limited improvements to the makespan can be attributed to the limitations regarding the explored parameter combinations, thus neglecting potential candidates for counterfactual explanations that differ in more than one feature from its original. NSGA-II demonstrates superior performance, particularly at 200 parameter combinations, achieving a peak average improvement of 71.8 min. The standard deviation is notably higher, ranging from 23.3 to 29.7 min, indicating a comparatively higher variability of identified counterfactuals.

Operation Improvement Regarding operation processing times, random search exhibits improvements ranging from 26.8 to 27.7 min on average, with a maximum standard deviation of 3.6 min, and is slightly outperformed by NNCE, with improvements ranging between 29.3 and 30 min. NSGA-II yields the highest and lowest improvements to the operation processing time, with values of 42.3 min for 200 parameter combinations and 25.3 min for 50 parameter combinations. The stability of NSGA-II highly depends on the number of parameter combinations, with standard deviations ranging from 12.1 min for 50 parameter combinations to 5.9 min for 200 parameter combinations. The difference in stability between makespan and processing time improvements of NSGA-II can be attributed to the random sampling of the starting population as well as the population. With the increasing number of explored parameter combinations, the tournament structure of NSGA-II contributes to retaining previously identified solutions from one generation to the next, thus yielding the highest improvements of the three algorithms.

Schedule Distance The comparative analysis of the schedule distances reveals a notable variation across different approaches. Random search and NSGA-II exhibit relatively higher schedule distances, with values ranging from 7.5 to 9.0. This indicates a significant deviation between the original and altered schedules generated through these methodologies. The accompanying standard deviations, ranging from 0.4 to 2.5, suggest variability in the consistency of these approaches, with random search displaying marginally better figures, implying a slightly more consistent performance compared to NSGA-II. The NNCE approach demonstrates superior performance in minimizing schedule distances, with the lowest average distance being 6.23 and the highest at 6.37, coupled with a more constrained standard deviation span of 0.587 to 1.22.

Operation Distance In terms of operation distance, there's a discernible differentiation in performance across the methodologies. Random search and NSGA-II again present comparable figures, with operation distances in the proximity of 0.77. However, it is noteworthy that NNCE outperforms

Table 5 Comparative analysis of random search, NNCE, and NSGA-II for counterfactual explanation search. The schedule to be optimized consisted of five jobs with a total amount of 24 operations. Each approach was executed ten times in order to mitigate outliers, with the table displaying averaged results as well as corresponding standard deviations

	Random search			NNCE			NSGA-II		
Parameter Combinations	50	100	200	50	100	200	50	100	200
Makespan	41.3	47.3	42.4	26.3	25.8	25.9	65.6	64.6	**71.8**
Improvement[a]	±6.5[b]	±5.6	±3.3	±5.66	±3.52	±2.18	±27.2	±23.2	±29.7
Operation	26.8	27.7	27.3	29.9	30.0	29.3	25.3	34.7	**42.3**
Improvement[a]	±3.6	±3.7	±2.8	±4.48	±4.7	±1.87	±12.1	±8.6	±5.9
Schedule	8.0	9.0	8.2	6.37	6.26	**6.23**	8.4	7.5	8.0
Distance	±1.4	±0.5	±0.4	±0.74	±1.22	±0.587	±2.5	±2.2	±2.2
Operation	0.77	0.83	0.81	**0.5**	**0.5**	**0.5**	0.77	0.71	0.73
Distance	±0.03	±0.04	±0.02	±0	±0	±0	±0.08	±0.1	±0.09
Identified	23.4	45.2	95.1	25.1	48.3	**98.6**	4.3	6.7	10.8
Explanations	±3.4	±4.0	±9.1	±2.4	±4.4	±4.6	±0.5	±0.8	±1.2
Excluded	5.1	7.3	21.8	5.5	10.7	21.3	1.5	1.4	2.0
Explanations[c]	±1.8	±3.6	±4.2	± 1.9	± 3.9	± 2.3	±0.5	±0.5	±1.0
Computation	5.81	11.07	21.9	5.87	11.33	22.12	6.54	12.55	24.01
Time[d]	±0.04	±0.05	±0.27	±0.11	±0.34	±0.23	±0.32	±0.89	±0.76

[a]improvements in minutes
[b]standard deviation
[c]explanations without makespan improvement
[d]computation time in seconds, without the costs of data format changes

both random search and NSGA-II in this metric, consistently achieving operation distances as low as 0.5 without any standard deviation. This performance of NNCE can be attributed to its strategic limitation to single variable value iterations, which inherently restricts the scope of alteration and thus maintains a closer alignment with the original operation parameters. This precision enables NNCE to not only generate counterfactuals that are easier to achieve but also ensure a higher fidelity to the original operational setup. NSGA-II, while showing a slight improvement over random search, still falls short of the benchmark set by NNCE, suggesting that despite NSGA-II's advancements in exploring multiple parameter combinations, it does not necessarily translate to a proportionate enhancement in maintaining operation proximity. The analysis elucidates the distinct capabilities and limitations of each approach in terms of maintaining schedule and operation proximity in the generation of counterfactual explanations. NNCE's methodology emerges as particularly effective in achieving closer approximations to the original configurations, thereby offering a compelling option for generating counterfactuals with minimized deviations.

Number of Identified Explanations The analysis of the number of identified explanations demonstrates a significant variation across the different algorithms, with random search and NNCE notably outperforming NSGA-II. Specifically, at the parameter combination of 200, Random Search and NNCE identified an average of 95.1 and 98.6 explanations, respectively. These figures significantly exceed the performance of NSGA-II, which identified far fewer explanations, respectively averaging 4.3, 6.7, and 10.8 for 50, 100, and 200 explored parameter combinations. The standard deviation associated with these findings, ranging from 0.5 to 9.1, indicates variability in the consistency of explanation identification across the algorithms, with NNCE demonstrating a notably tighter range of deviation, thus suggesting a higher consistency in the amount of identified counterfactual explanations. NSGA-II's lower performance in this regard is due to its inherent population size limitations, which constrain its ability to identify a broader array of explanations within the same parameter combination scope. As an example, with 200 parameter combinations being explored over ten populations, the population size limit constraints NSGA-II to a maximum of 20 unique counterfactual explanations.

Number of Excluded Explanations Approximately 20 to 25% of the explanations identified by the examined algorithms did not result in an improvement of the schedule makespan, indicating a noteworthy proportion of explanations that, while potentially optimizing processing times for operations, do not contribute to the overall efficiency of the schedule. This finding highlights the complexity of job scheduling challenges, where not all operational improvements necessarily lead to better schedule outcomes. The number of excluded explanations, ranging from an average of 5.1 to 21.8 with a standard deviation of 0.5 to 4.2, suggests that a significant fraction of operational adjustments, despite being identified as potential improvements, do not directly

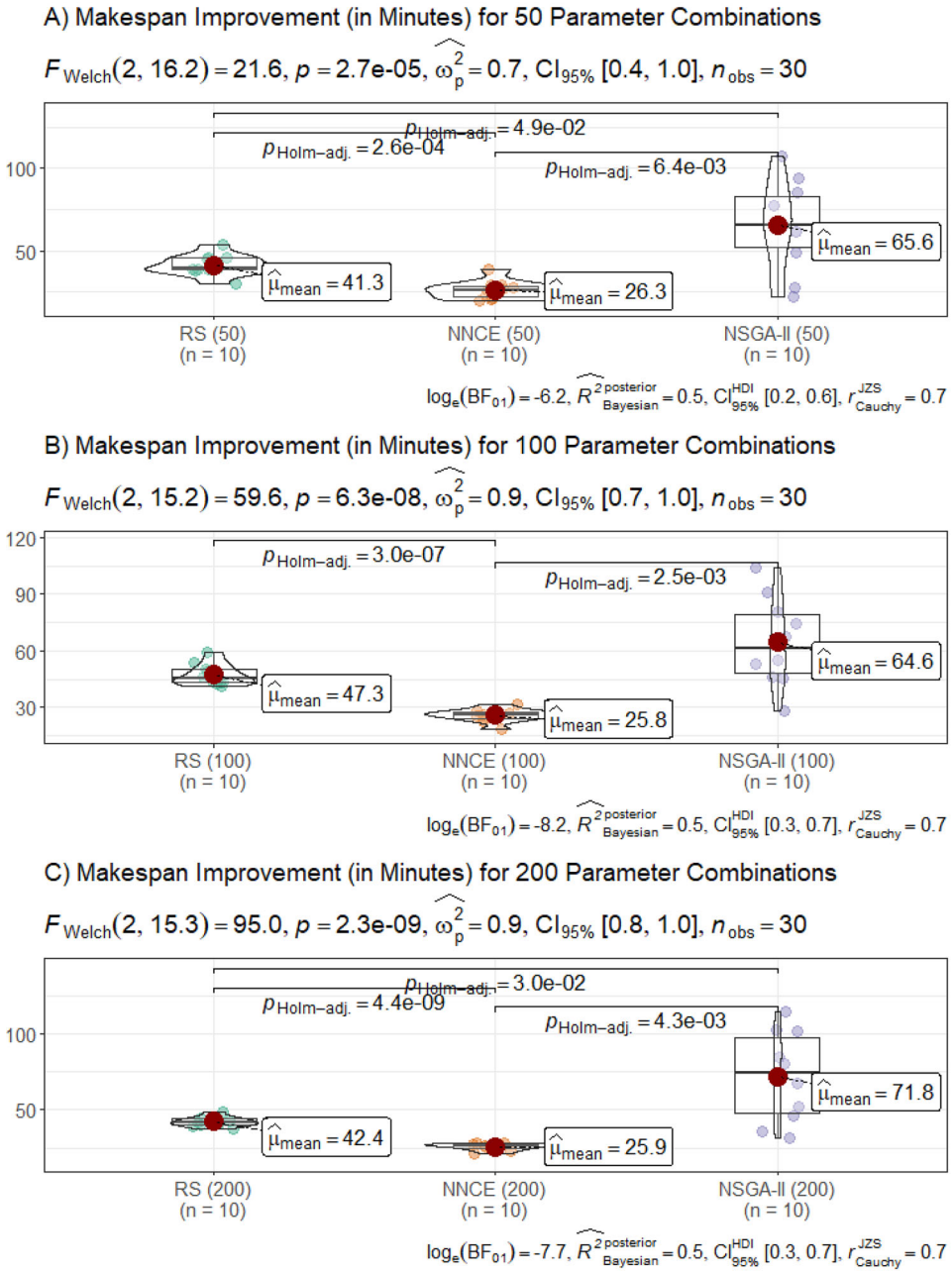

Fig. 4 Significance test for makespan improvements of altered schedules for identified counterfactual explanations, comparing random search, NNCE, and NSGA-II

translate into enhanced schedule performance. This discrepancy could provide valuable insights into the granularity of job scheduling processes, particularly in understanding the minimum necessary operational improvements that could lead to meaningful schedule optimizations. It reflects the inherent challenge in aligning operational modifications with overarching scheduling objectives, emphasizing the need for a careful consideration of both operational and schedule-level impacts in the pursuit of optimization strategies.

For further performance evaluation, pairwise comparisons of the three search approaches were performed to identify significant differences. In particular, Welch's *t*-test was performed with a *p*-value threshold of 0.05 as well as a Bayesian analysis, contrasting different search approaches across explored parameter combinations for improvements in makespan, operation processing time, the number of identified explanations and computation time. Figures 4, 5, and 6, and respectively illustrate the results of the significance tests, each divided into three subplots (A), (B), and (C) for the parameter combinations 50, 100, and 200, respectively. Figure 4A) visualizes the makespan improvement for each of the ten iterations of the random search, NNCE and NSGA-II via boxplots and violin plots. The textual descriptions above the plot detail the test method (Welch), the degrees of freedom for

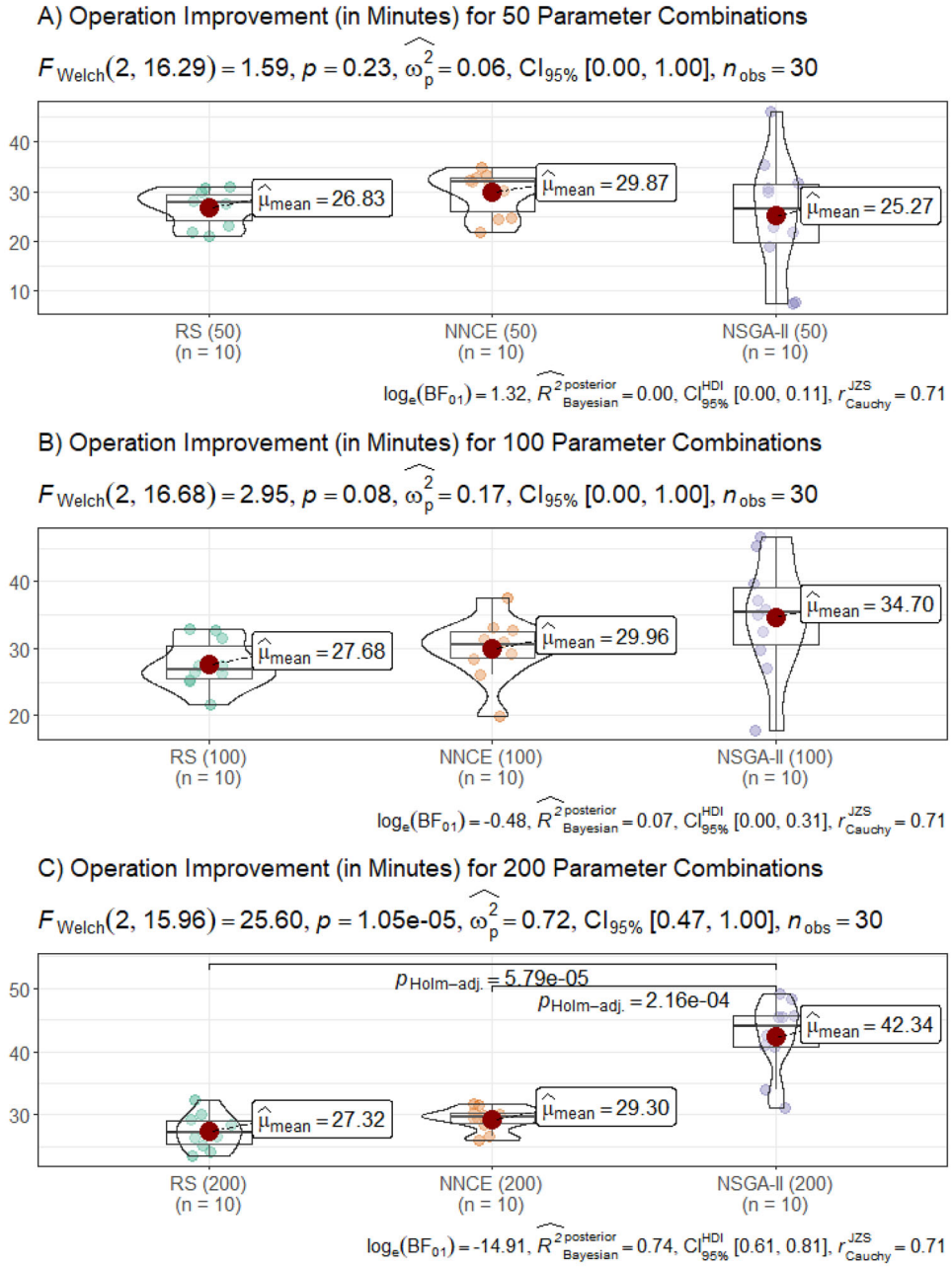

Fig. 5 Significance test for processing time improvements of operations for identified counterfactual explanations, comparing random search, NNCE, and NSGA-II

the t-statistic (2, 16.2), the value of the statistic (21.6), the *p*-value (0.000027), the effect size type (partial omega-squared) and the estimate (0.7), the confidence interval ([0.4, 1.0]) and the total number of observations (30), while the descriptions below the plot document the Bayes analysis with the natural logarithm of the Bayes Factor BF_{01} (−6.2), the posterior type and its estimation (0.5) as well as the credible interval ([0.2, 0.6]), the prior type (Cauchy) and value (0.7). The Figs. 5 and 6 illustrate results in an analogous manner. The results of the significance analysis further confirm previous observations: NSGA-II outperforms random search and NNCE by a significant margin with regard to average improvements in makespan and operation processing time, particularly as the number of explored parameter combinations increases. Differences between random search and NNCE are significant for improvements in the schedule makespan and can be attributed, as previously stated, to the limitations imposed by the nearest neighbor search of NNCE. For the number of identified explanations, random search and NNCE significantly outperform NSGA-II across any number of explored parameter combinations, due to the restricted population size of the NSGA-II approach.

In each of the three test categories above, the disparities between the examined approaches became more pronounced

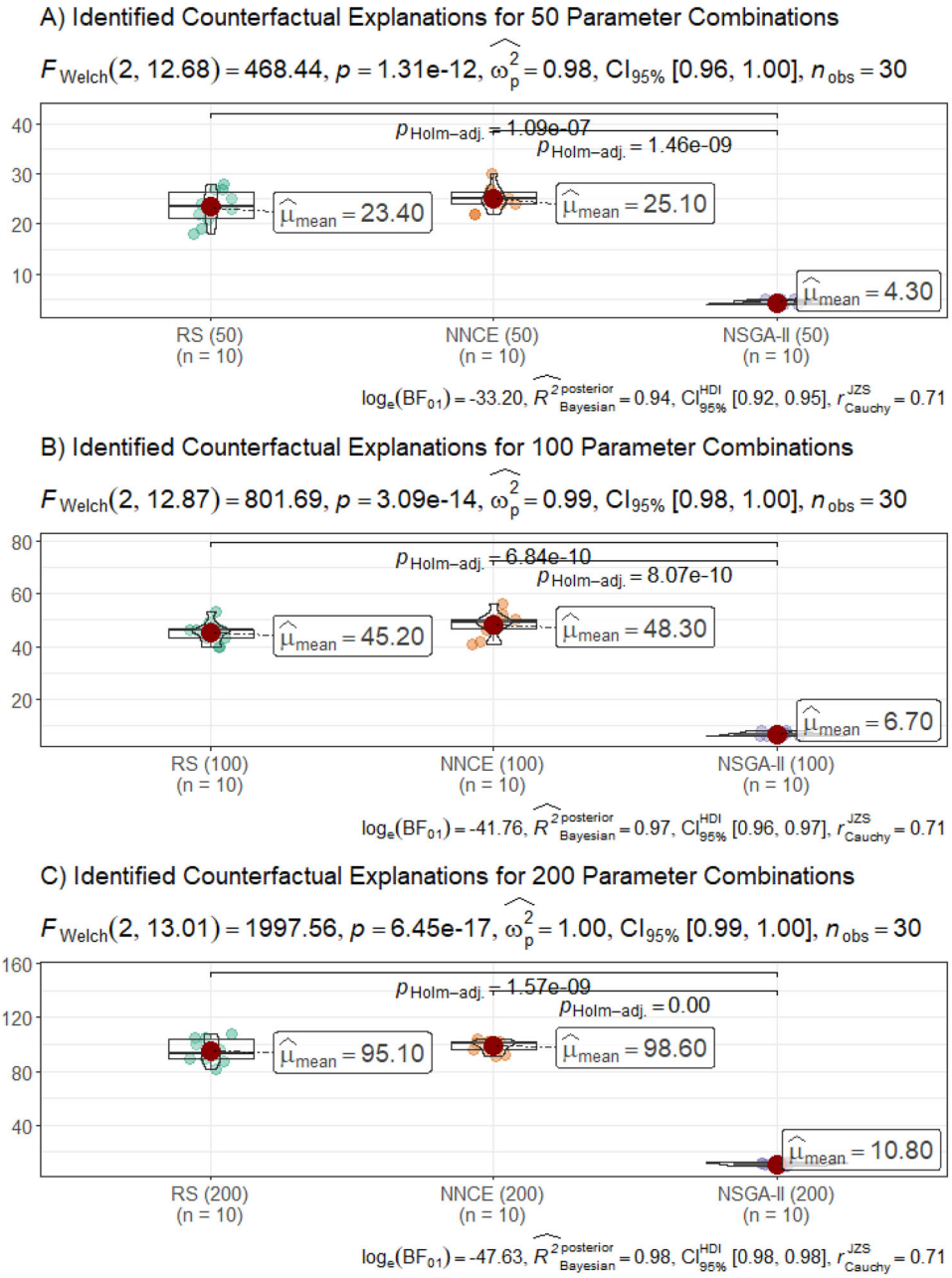

Fig. 6 Significance test for the number of identified counterfactual explanations, comparing random search, NNCE, and NSGA-II

as more parameter combinations were investigated. These results demonstrate that NSGA-II exhibits superior performance with regard to identifying counterfactual explanations with improved schedule makespans and operation processing times, particularly at higher levels of explored parameter combinations. Its effectiveness in optimizing complex schedules, however, is accompanied by higher variability, as evidenced by the standard deviation values. This suggests that while NSGA-II can achieve significant improvements, its outcomes may be less predictable compared to random search or NNCE. Furthermore, random search and NNCE significantly outperformed NSGA-II with regard to the number of identified solutions, particularly at higher parameter combinations. This could be attributed to the exhaustive nature of random search in exploring the solution space. NNCE provides a slightly higher amount of identified counterfactuals than random search, which can be attributed to its search space being restricted to alternatives similar to the original, thus resulting in a minimally larger amount of potentially viable solutions. However, the number of identified solutions that fail to improve the schedule makespan suggests that random search and NNCE both exhibit diminishing returns, trading the quality of the identified counterfactuals for quantity, emphasized by NSGA-II providing the

highest improvements regarding makespan and operation processing times. Subsequent statistical analyses corroborated the superior performance of NSGA-II in refining makespan and operation processing times as the number of explored parameter combinations increased. Our findings elucidate the nuanced strengths and limitations of each approach, presenting NSGA-II as a notably effective strategy for optimizing complex schedules through counterfactual explanations, albeit with an inherent variability that warrants consideration.

Comparative Evaluation Regarding Computation Times

Evaluating computational costs is essential for optimizing algorithm efficiency and ensuring their practical applicability. It allows for the identification of resource-efficient algorithms and facilitates a balance between outcome quality and computational expenditure. This assessment is particularly vital in complex tasks, where the computational intensity of algorithms can significantly impact their viability and effectiveness, guiding the selection toward solutions that effectively balance accuracy, speed, and resource demands. For this purpose, the computation times of the random search, NNCE, and NSGA-II approaches were measured and evaluated, with the results being presented in Table 5 and Fig. 7. In light of the utilized *h2o* library (see the "Dataset Overview and Tools" section, the computational costs of transforming tabular data from R-data-frames to h2o-data-frames and vice versa turned out as costly and the intricacies of implementing methods from this library were deemed a disruptive factor in comparing the three approaches fairly. Hence, for the comparative evaluation of computation times, we isolated the data format transformation steps that are specific to the implemented algorithm and excluded their computation time, so that the analysis results do not hinge on the chosen ML packages and their implementation.

Regarding computation time, all three approaches exhibit a linear increase proportional to the amount of explored parameter combinations, with random search and NNCE showing similar results slightly in favor for random search. In particular, the exploration of 50, 100, and 200 parameter combinations takes an average of 5.81 s, 11.07 s, and 21.9 s respectively for a random search, and is closely followed by NNCE. NSGA-II, however, requires more computational resources, exhibiting an approximately 10% increase in computation time across all explored parameter combinations. This increase in computational cost can be attributed to the iterative generation and evaluation of counterfactual explanations generated by NSGA-II as opposed to the straightforward nature of the random search and NNCE approaches. Furthermore, it should be noted that the measured computation times excluded the time necessary for any data format transformations, which are specific to the implementation of the chosen ML model. As an example, random search is able to sample 200 parameter combinations from the feature space and proceed to the scoring by the chosen ML model, whereas NSGA-II integrates the ML scoring at each generation before the tournament step. The significance tests presented in Fig. 7 further confirm that the computational costs associated with random search and NNCE vary noticeably from NSGA-II, showing significant differences across all search depths. NSGA-II, with its sophisticated evolutionary operations, incurs higher computational demands due to the complexity of its non-dominated sorting and diversity preservation mechanisms. Random Search, while exhaustive, has relatively lower computational requirements, stemming from its simplicity. NNCE occupies a middle ground, as its nearest neighbor search involves moderate computational efforts, balanced by its targeted exploration. Thus, the selection of an algorithm must consider the trade-off between computational efficiency and the quality of optimization outcomes, highlighting the necessity of aligning computational resources with optimization objectives.

Comparative Evaluation Regarding Counterfactual Desiderata

This section delineates a comparative analysis among the random search, NNCE, and NSGA-II methodologies, in the context of generating counterfactual explanations that adhere to the specific desiderata of validity, sparsity, plausibility, actionability, proximity, and diversity (see the "Counterfactual Explanations Generation and Evaluation" section).

Random Search The random search method possesses the capability to generate valid counterfactuals due to its exhaustive exploration of the search space. Our implementation restricted variable alterations to machines and corresponding resources viable for the operation, however, this approach does not inherently guarantee the validity or actionability of the counterfactuals produced, as it lacks a targeted mechanism. This may result in the generation of counterfactuals that suggest changes beyond the control or capacity of the decision-maker, thereby reducing their practical value. Sparsity is not a primary consideration within the random search approach and may yield counterfactuals that involve numerous changes from the original instance, potentially complicating their interpretation and diminishing their usefulness. Given its disregard for domain-specific knowledge and constraints, this approach runs the risk of generating counterfactuals that are implausible within the context of the application domain and can limit the applicability of the explanations in real-world scenarios. Disregarding the limitations imposed by our implementation, this method con-

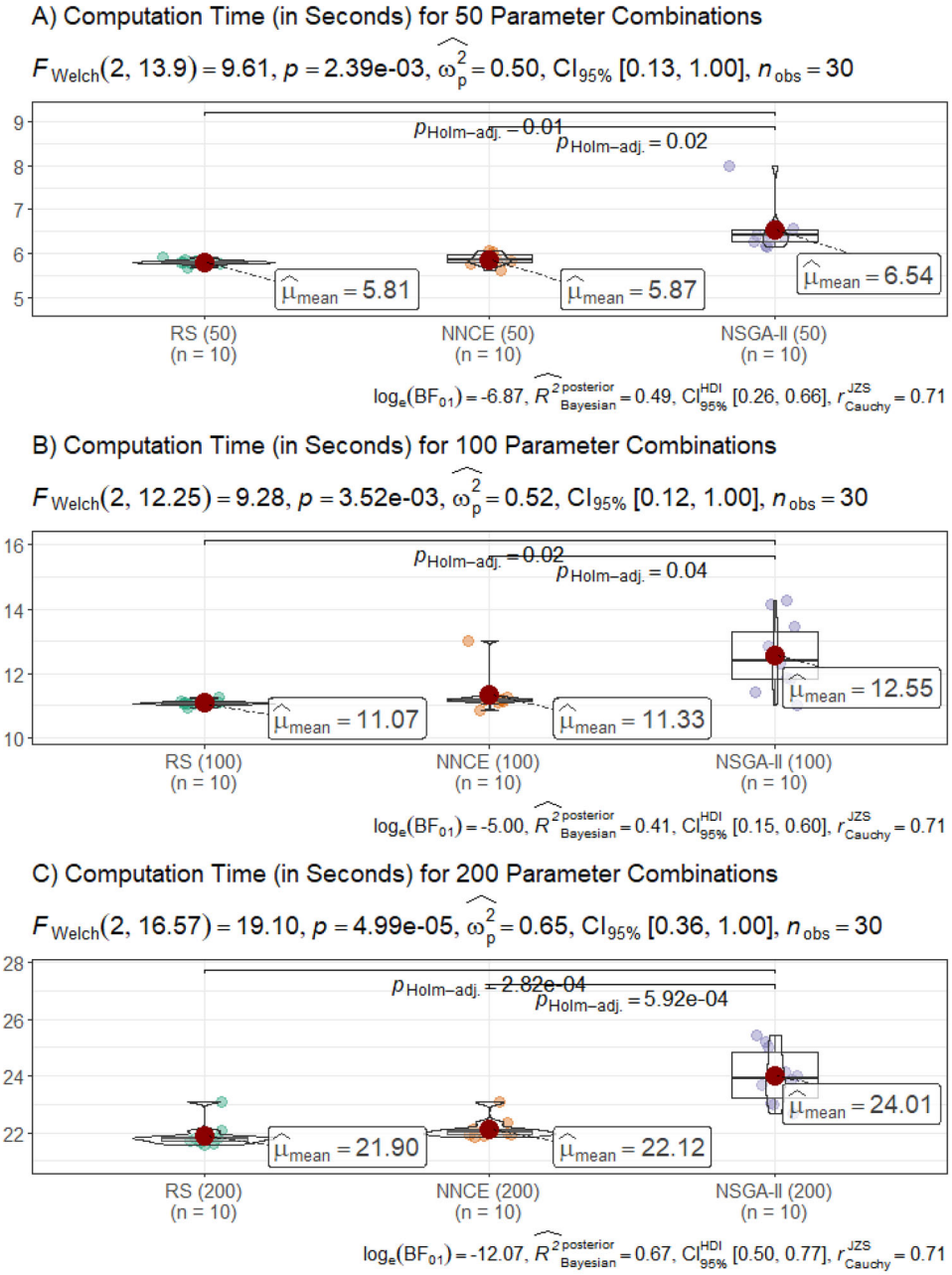

Fig. 7 Significance test for the computation time, comparing random search, NNCE, and NSGA-II

ventionally lacks a systematic approach to ensure proximity between the original instance and the generated counterfactuals. As a result, the counterfactuals may exhibit significant deviations from the original instance, undermining their applicability and transferability to other application scenarios. While this approach is capable of producing a diverse array of counterfactuals due to its non-deterministic nature, it lacks a structured methodology to ensure that this diversity translates into a meaningful variety of actionable and plausible explanations.

NNCE The NNCE apporach is explicitly designed to increase the validity of counterfactual explanations by focusing on the identification of nearest neighbors that lead to a change in the decision outcome. By prioritizing the identification of the closest instances that result in a different decision, NNCE naturally encourages the generation of sparse counterfactuals in the proximity of the original instance. This focus on minimal changes supports the creation of explanations that are simpler and more interpretable. The reliance on nearest neighbors further increases the likelihood of gen-

erating plausible counterfactuals, enhancing their credibility, and utility. Similar to our implementation of random search, variable alterations were limited to machines and resources able to operate them, offering actionable changes. However, the leveraged nearest-neighbor methodology does not ensure actionability by default. Although NNCE is capable of generating counterfactuals that are close to the original decision boundary, its focus on nearest changes may limit the diversity of the counterfactuals produced. This concentration may overlook other valid and potentially insightful counterfactuals, as indicated with regard to the analysis of average makespan improvements in the "Comparative Evaluation Regarding Objectives" section.

NSGA-II NSGA-II is able to effectively generate valid counterfactuals by optimizing across multiple objectives, ensuring that the counterfactuals produced are capable of changing the model's predictions in a meaningful way. The configuration of this approach allows for the prioritization of sparsity and actionability through its fitness function, similar to the implementation of random search and NNCE, where alterations were constrained to machines suitable for the specified operation type, in addition to the resources capable of managing these machines. NSGA-II aims to maintain proximity between the original instance and the counterfactuals through its consideration of proximity as an optimization objective. This focus supports the generation of counterfactuals that represent minimal and feasible departures from the original instance. By nature of its evolutionary algorithm framework, NSGA-II excels in supporting diversity among the generated counterfactual explanations. This algorithm produces a range of Pareto optimal solutions, offering a broad spectrum of alternative explanations and insights into the decision-making process. However, this approach lacks the implementation of domain-specific knowledge into its optimization process, thus necessitating a manual examination of the plausibility of the generated counterfactuals.

Comprehensive Exploration of Multi-objective Counterfactuals

To address **RQ2** regarding the quality and applicability of identified counterfactuals, two exemplary counterfactual explanations will be examined. An excerpt of all identified alternative operation settings identified via NSGA-II for 200 explored parameter combinations is depicted in Fig. 8, with Fig. 8A) presenting the found counterfactuals in a tabular manner, providing information about relevant variable values as well as operation and makespan improvements.

Figure 8B) depicts the original variable values of the altered operation for context, with Fig. 8C) providing a textual description of the relevant alterations and predicted improvements to the operation and job schedule. Figure 9 depicts the Pareto front constructed by NSGA-II with a 3D-visualization of a scatter plot along the axis of operation ("Operation_Improvement"), makespan improvement ("Schedule_Improvement") and the distance between the original and altered operation ("Operation_Distance"), with the distance between the original and altered schedule being depicted through a gradient color-coding. Each point represents a counterfactual explanation and its position within the plot is determined by its quality regarding the objectives of the counterfactual search (see Definition 6).

The first example demonstrates an improvement in the processing time for the *Flanging* activity performed on machine "C" using worker with the identifier "137" (see Fig. 3) of job 11679 by allocating it to machine "B" and worker "775" (see Fig. 10). Although the improvement for the operation is 59 min, significantly reducing the predicted processing time from 227 to 168 min, the schedule cannot be optimized in a way that reduces the total makespan. The completion times for jobs 116,591, 112,996, and 117,583 were moved forward, with the completion time of job 11679 being postponed. Although slight changes are observed in the schedules, the sequencing of operations remains largely unchanged.

The second example demonstrates the maximum possible improvement with the change of a single operation while disregarding substantial changes in the scheduling of operations, changing the machine for the first operation of job 11679 from *Dishing Press* "A" to "D" and the responsible worker from "405" to "125" (see Fig. 11). This change improved the predicted duration of the operation by 29 min (from 203 min to 174 min) as well as the schedule makespan by 192 min, moving the completion times of all jobs forward as a consequence. When compared with the first example, the reduction in operation processing time is comparably small, however, the position of the operation within the production environment allows for more drastic changes to the process schedule and, consequently, radical improvements with regards to the makespan.

These examples highlight the complexities of multi-objective optimization in the scheduling problem and stress the importance of comprehensive exploration when implementing optimization strategies. The contrast between the two scenarios underscores that while targeted improvements in individual operations can be beneficial, their overall impact on the production schedule can vary significantly. This neces-

A)

Identified Counterfactual Explanations for Schedule with a Makespan of 1003 minutes:

Job ID	Operation Position	Operation	Diameter Blank	Weight	...	Machine	Resource ID	Predicted Duration	Operation Improvement	Makespan Improvement
111679	1	Dishing Press	940	17	...	D	125	174.13661	-29.0	-192.1
111679	5	Edge Deburring	940	17	...	D	633	214.02923	-107.2	-123.8
111679	5	Edge Deburring	940	17	...	B	103	280.42923	-40.8	-123.8
111679	5	Edge Deburring	940	17	...	B	161	291.92923	-29.3	-123.8
111679	5	Edge Deburring	940	17	...	A	648	217.42923	-103.8	-103.6
114735	4	Cleanup	1690	144	...	A	110	79.90274	-117.3	-94.3
112996	1	Dishing Press	1380	48	...	B	114	52.10314	-27.0	-76.4
111679	4	Flanging	940	17	...	C	114	166.79956	-60.4	-44.7
111679	1	Dishing Press	940	17	...	A	123	159.13661	-44.0	-44.0
111679	4	Flanging	940	17	...	B	160	138.29956	-88.9	-28.6

B)

Original Operation

Job ID	Operation Position	Operation	Diameter Blank	Weight	...	Machine	Resource ID	Predicted Duration
111679	5	Edge Deburring	940	17	...	C	161	321.2292

C) The selected counterfactual changes the Machine from C to D and changes the Resource with ID 161 to the Resource with ID 633, leading to an operation processing time of 214 minutes. This alteration reduces the original prediction of 321.2 minutes by 107.2 minutes.

This alteration also leads to a schedule makespan of 879.7 minutes, reducing the original makespan prediction of 1003.5 minutes by 123.8 minutes.

Regarding makespan improvements, no other counterfactual exist that can further decrease the makespan by iterating parameters of this operation.

Fig. 8 Exemplary extract of **A** the list of identified counterfactual explanations using NSGA-II with a search depth of 200 parameter combinations, showing the top 10 counterfactual explanations in terms of schedule and operation distance as well as improvements in schedule makespan and processing times, **B** the original instance corresponding to the counterfactual that was selected for further examination, and **C** a verbalization of the selected counterfactual explanation. The row of the selected counterfactual and its corresponding original are highlighted yellow, indicating alterations to the machine and resource with orange color. Processing times and makespan as well as their improvements are highlighted using blue color-coding

sitates a more thorough and explorative approach to the optimization process.

Discussion

Our study has demonstrated a novel approach to leveraging counterfactual explanations within the context of JSSP, an intersection of AI, operations research (OR), and process mining. This research has successfully combined complex aspects of decision-making by using a multi-objective optimization framework, specifically the NSGA-II. This has shown how even small changes in operational parameters can have a big effect on the overall efficiency of a system. This integration not only amplifies the potential of AI and OR in process optimization but also enriches the field of process mining with a deeper understanding of how individual process steps influence the larger system.

In recognizing the specialized focus of our investigation on generating counterfactuals for production scheduling scenarios, we also highlight the extensive applicability of our methods across a broad array of domains. These diverse areas-ranging from supply chain optimization and healthcare planning optimization to capacity planning, energy market modeling, transport planning, and logistics optimization, as well as integrated sales and operations planning-benefit from the synergistic integration of predictive analytics and optimization algorithms. Our research contributes significantly to these fields by providing a methodological framework for generating valuable counterfactual insights, which is not only relevant but indispensable for enhancing decision-making processes. Therefore, while our work may originate from a

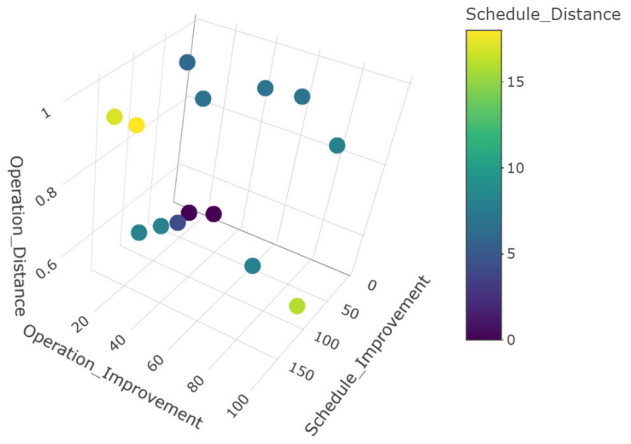

Fig. 9 Snippet of the 3D-visualization of the multi-dimensional Pareto front regarding operation and schedule distances as well as corresponding improvements, yielded by NSGA-II with a search depth of 200 parameter combinations, depicting the scores of identified counterfactual explanations in the form of points, positioned and color-coded accordingly

niche perspective, its implications are far-reaching, underscoring the transferability and critical importance of our approach in various interdisciplinary contexts. This extension of our research into other domains emphasizes the universal applicability and potential impact of our findings, illustrating that, despite its specialized nature, our work serves as a foundational pillar for advancing practices in numerous other sectors where predictive modeling and optimization converge.

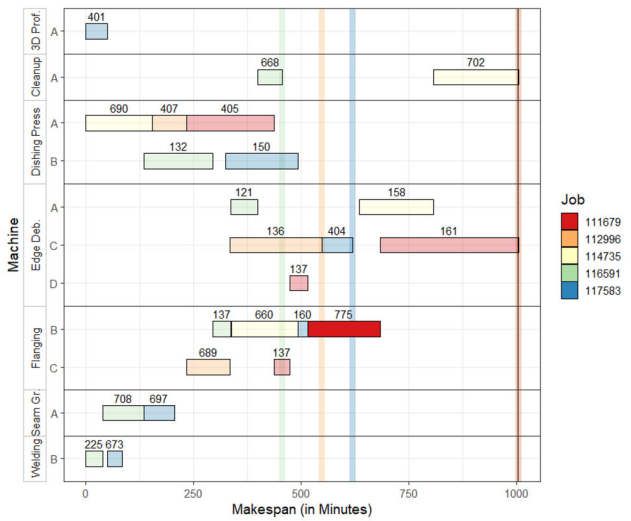

Fig. 10 Alternative schedule without makespan improvement, highlighting *Flanging* performed on the alternative machine "B" by worker "775" after alteration. Although the processing time of the highlighted operation decreased by 59 min and several jobs terminated earlier, the schedule makespan could not be improved

The relevance of our study to decision augmentation is particularly significant, offering a robust framework that enhances the capabilities of decision-makers in complex environments. By integrating counterfactual explanations with multi-objective optimization, we provide a tool that augments human decision-making with deep, AI-driven insights. This approach allows decision-makers to understand not only what decision might be optimal but also why it is so by revealing the intricate interdependencies and potential impacts of various choices. It empowers them to make informed decisions that consider a wide range of scenarios and outcomes, ultimately leading to more effective, efficient, and responsible outcomes. This augmentation is especially valuable in sectors where the complexity and volume of data exceed human capacity to analyze, making AI an essential partner in the decision-making process.

In the context of industrial analytics, our study's approach to responsible AI is also relevant, offering a framework that aligns with the industry's growing need for transparent, ethical, and accountable AI systems. By applying counterfactual explanations within a multi-objective optimization setting, we provide a means for industrial analytics to not only make AI-driven decisions transparent but also ensure these decisions are ethically sound and accountable. In an industrial setting, where decisions often have substantial economic and operational implications, understanding the rationale behind AI predictions and recommendations is crucial. Our approach allows for a clear exposition of how different variables and scenarios impact outcomes, making AI systems more interpretable and trustworthy to engineers, managers, and stakeholders. This transparency is vital for maintaining confidence in AI-driven industrial processes. Additionally, our methodology's emphasis on ethical considerations in AI decision-making ensures that industrial analytics systems are designed to be fair, avoiding biases that could lead to inefficiencies or unfair practices. This aspect is particularly important in industries where AI decisions might impact workforce management, environmental compliance, or safety protocols. Lastly, by enhancing the accountability of AI systems in an industrial analytics setting, our study ensures that AI-driven decisions can be audited and improved upon. This aspect is crucial for industries to adapt and evolve their AI systems responsibly, ensuring continuous improvement and alignment with industry standards and regulations. Therefore, in the sphere of industrial analytics, our approach not only advances the technical capabilities of AI but also ensures these advancements are responsibly aligned with the broader goals and values of the industry.

Furthermore, the study emphasizes the relevance of system thinking in manufacturing and process management. By considering the manufacturing process as an interconnected system, our approach enables a comprehensive evaluation of the implications of individual changes on the entire system.

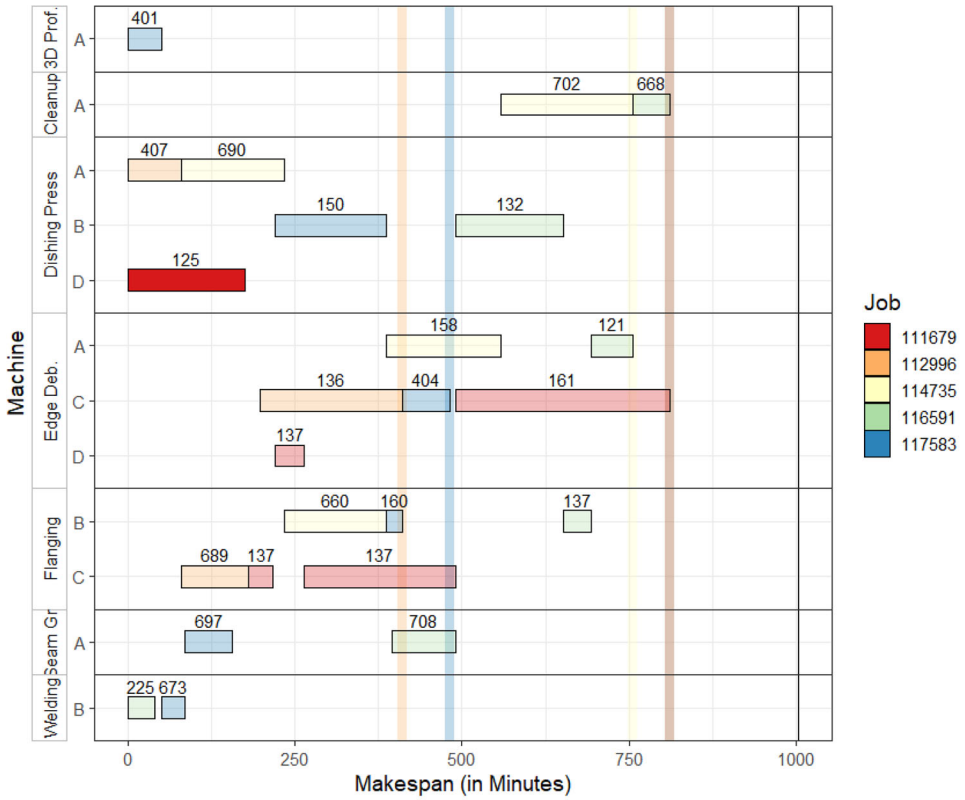

Fig. 11 Alternative schedule, highlighting the activity performed on the alternative *Dishing Press* "C" by worker "125" after alteration. With a comparably small reduction in the processing time of the highlighted operation, the schedule makespan was improved by 192 min with several jobs terminated earlier compared to the original schedule

This systemic perspective is crucial in identifying optimal solutions that enhance individual operations and overall process efficiency. It encourages a shift from focusing solely on isolated process improvements to a more integrated approach that considers the cumulative effect of these improvements on the entire system. In this regard, incorporating the granular computing paradigm is pivotal in shaping our analytical approach and aligning it with the composition of process mining data. This concept enables nuanced navigation and consideration of objectives and factors across different system levels when identifying counterfactual explanations, facilitating the transition from local optimizations to systematic improvements. The adaptability and structured perspective provided by granular computing are particularly congruent with the nature of process mining data, which inherently comprises multi-level, interconnected information. By employing Granular Computing, we effectively segment the objectives for identifying counterfactuals within the complex scheduling problem into more manageable and interpretable units, ensuring enhancements contribute meaningfully to the system's overall efficiency. This methodological alignment enhances our capability to dissect and interpret process mining data in an insightful and actionable manner. Consequently, our application of granular computing fosters a framework for identifying impactful counterfactual explanations and amplifies the potential for systemic improvements within the intricate dynamics of job-shop scheduling. The paradigm's emphasis on considering the granularity of data and system components ensures our findings are comprehensive and relevant, paving the way for holistic advancements rather than isolated improvements. Through this lens, granular computing serves as a cornerstone of our study, guiding our exploration of process mining data toward achieving significant, system-wide enhancements.

Conclusion

In conclusion, this research introduced a sophisticated method for generating counterfactual explanations in complex scheduling scenarios, specifically in the context of the JSSP. The core methodology integrated a predictive ML model with MILP for initial schedule generation, followed by the application of the NSGA-II for multi-objective optimization. The results of the study revealed the effectiveness of the NSGA-II approach in producing robust counterfactual explanations, particularly in improving schedule makespan and operation processing times. This was notably more efficient than the alternative counterfactual explanation method, especially at higher levels of parameter combinations. Furthermore, the exploration of specific counterfactual scenarios underscored the critical role of operation placement within the scheduling environment. It was observed that improvements in individual operations could

have significantly different impacts on the overall schedule, depending on their interaction with other operations and their position in the process flow. Overall, the study successfully demonstrated a comprehensive framework for enhancing both individual operation efficiencies and system-wide performance in scheduling environments. The insights gained from this research not only contribute to the field of explainable AI but also provide valuable guidelines for future applications in optimization and scheduling tasks.

Author Contribution Each author contributed equally to writing and reviewing the manuscript.

Funding Open Access funding enabled and organized by Projekt DEAL. This research was funded in part by the German Federal Ministry of Education and Research under grant number 01IS21006B (project ExPro).

Data Availability No datasets were generated or analyzed during the current study.

Declarations

Ethical Approval This article does not contain any studies with human participants or animals performed by any of the authors.

Competing Interests The authors declare no competing interests.

References

1. Arrieta AB, Díaz-Rodríguez N, Del Ser J, Bennetot A, Tabik S, Barbado A, García S, Gil-López S, Molina D, Benjamins R, et al. Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI. Inf Fusion. 2020;58:82–115.
2. Doshi-Velez F, Kim B. Towards a rigorous science of interpretable machine learning. arXiv:1702.08608 [Preprint]. 2017. Available from: http://arxiv.org/abs/1702.08608.
3. Wachter S, Mittelstadt B, Russell C. Counterfactual explanations without opening the black box: automated decisions and the GDPR. Harv JL & Tech. 2017;31:841.
4. Guidotti R. Counterfactual explanations and how to find them: literature review and benchmarking. Data Min Knowl Disc. 2022;1–55.
5. Korikov A, Shleyfman A, Beck C. Counterfactual explanations for optimization-based decisions in the context of the GDPR. In: ICAPS 2021 Workshop on Explainable AI Planning. 2021.
6. Lerouge M, Gicquel C, Mousseau V, Ouerdane W. Counterfactual explanations for workforce scheduling and routing problems. In: 12th International Conference on Operations Research and Enterprise Systems. SCITEPRESS-Science and Technology Publications; 2023. pp. 50–61.
7. Bargiela A, Pedrycz W. Granular computing. In: Handbook on Computer Learning and Intelligence: Volume 2: Deep Learning, Intelligent Control and Evolutionary Computation. World Scientific; 2022. pp. 97–132.
8. Gunning D, Stefik M, Choi J, Miller T, Stumpf S, Yang G-Z. XAI–explainable artificial intelligence. Sci Robot. 2019;4(37):7120.
9. Angelov PP, Soares EA, Jiang R, Arnold NI, Atkinson PM. Explainable artificial intelligence: an analytical review. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. 2021;11(5):1424.
10. Wexler J, Pushkarna M, Bolukbasi T, Wattenberg M, Viégas F, Wilson J. The what-if tool: interactive probing of machine learning models. IEEE Trans Visual Comput Graphics. 2019;26(1):56–65.
11. Mothilal RK, Mahajan D, Tan C, Sharma A. Towards unifying feature attribution and counterfactual explanations: Different 1654 means to the same end. In: Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society. 2021. pp. 652–63.
12. Mothilal RK, Sharma A, Tan C. Explaining machine learning classifiers through diverse counterfactual explanations. In: Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency. 2020. 607–617.
13. Spreitzer N, Haned H, van der Linden I. Evaluating the practicality of counterfactual explanations. In: Workshop on Trustworthy and Socially Responsible Machine Learning, NeurIPS 2022. 2022.
14. Artelt A, Vaquet V, Velioglu R, Hinder F, Brinkrolf J, Schilling M, Hammer B. Evaluating robustness of counterfactual explanations. In: 2021 IEEE Symposium Series on Computational Intelligence (SSCI). IEEE; 2021. pp. 1–9.
15. Keane MT, Kenny EM, Delaney E, Smyth B. If only we had better counterfactual explanations: five key deficits to rectify in the evaluation of counterfactual XAI techniques. arXiv:2103.01035 [Preprint]. 2021. Available from: http://arxiv.org/abs/2103.01035.
16. Mittelstadt B, Russell C, Wachter S. Explaining explanations in AI. In: Proceedings of the Conference on Fairness, Accountability, and Transparency. 2019. pp. 279–88.
17. Kanamori K, Takagi T, Kobayashi K, Arimura H. DACE: distribution-aware counterfactual explanation by mixed-integer linear optimization. In: IJCAI. 2020. pp. 2855–62.
18. Karimi A-H, Barthe G, Balle B, Valera I. Model-agnostic counterfactual explanations for consequential decisions. In: International Conference on Artificial Intelligence and Statistics. PMLR; 2020. pp. 895–905.
19. Cheng F, Ming Y, Qu H. DECE: decision explorer with counterfactual explanations for machine learning models. IEEE Trans Visual Comput Graphics. 2020;27(2):1438–47.
20. Martens D, Provost F. Explaining data-driven document classifications. MIS Q. 2014;38(1):73–100.
21. Rathi S. Generating counterfactual and contrastive explanations using SHAP. arXiv:1906.09293 [Preprint]. 2019. Available from: http://arxiv.org/abs/1906.09293.
22. Dandl S, Molnar C, Binder M, Bischl B. Multi-objective counterfactual explanations. In: International Conference on Parallel Problem Solving from Nature. Springer; 2020. pp. 448–69.
23. Gomez O, Holter S, Yuan J, Bertini E. ViCE: visual counterfactual explanations for machine learning models. In: Proceedings of the 25th International Conference on Intelligent User Interfaces. 2020. pp. 531–5.
24. Schleich M, Geng Z, Zhang Y, Suciu D. GeCo: quality counterfactual explanations in real time. arXiv:2101.01292 [Preprint]. 2021. Available from: http://arxiv.org/abs/2101.01292.

25. Yao Y. Three perspectives of granular computing. Journal of Nanchang Institute of Technology. 2006;25(2):16–21.
26. Lin TY. Granular computing: structures, representations, and applications. In: Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing: 9th International Conference, RSFDGrC 2003, Chongqing, China, May 26–29, 2003 Proceedings 9. Springer; 2003. pp. 16–24.
27. Wang G, Yang J, Xu J. Granular computing: from granularity optimization to multi-granularity joint problem solving. Granular Computing. 2017;2:105–20.
28. Pedrycz W. Allocation of information granularity in optimization and decision-making models: towards building the foundations of granular computing. Eur J Oper Res. 2014;232(1):137–45.
29. Skowron A, Stepaniuk J, Swiniarski R. Modeling rough granular computing based on approximation spaces. Inf Sci. 2012;184(1):20–43.
30. Kumar MS, Gupta I, Panda SK, Jana PK. Granularity-based workflow scheduling algorithm for cloud computing. J Supercomput. 2017;73:5440–64.
31. Friedman JH. Greedy function approximation: a gradient boosting machine. Ann Stat 2001;1189–232.
32. Nelder JA, Wedderburn RW. Generalized linear models. J R Stat Soc Ser A Stat Soc. 1972;135(3):370–84.
33. Geurts P, Ernst D, Wehenkel L. Extremely randomized trees. Mach Learn. 2006;63:3–42.
34. Srinivas N, Deb K. Muiltiobjective optimization using nondominated sorting in genetic algorithms. Evol Comput. 1994;2(3):221–48.
35. Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput. 2002;6(2):182–97.
36. Mehdiyev N, Majlatow M, Fettke P. Quantifying and explaining machine learning uncertainty in predictive process monitoring: an operations research perspective. arXiv:2304.06412 [Preprint]. 2023. Available from: http://arxiv.org/abs/2304.06412.
37. Mehdiyev N, Majlatow M, Fettke P. Communicating uncertainty in machine learning explanations: a visualization analytics approach for predictive process monitoring. arXiv:2304.05736 [Preprint]. 2023. Available from: http://arxiv.org/abs/2304.05736.
38. Willmott CJ, Matsuura K. Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. Clim Res. 2005;30(1):79–82. Accessed 20 Mar 2023. Available from: https://www.int-res.com/articles/cr2005/30/c030p079.pdf.
39. Chai T, Draxler RR. Root mean square error (RMSE) or mean absolute error (MAE)?-arguments against avoiding RMSE in the literature. Geosci Model Dev. 2014;7(3):1247–50.
40. Delacre M, Lakens D, Leys C. Why psychologists should by default use Welch's t-test instead of Student's t-test. Int Rev Soc Psychol. 2017;30(1):92–101.
41. Patil I. Visualizations with statistical details: the 'ggstatsplot' approach. J Open Source Softw. 2021;6(61):3167.
42. Hedges LV. Estimation of effect size from a series of independent experiments. Psychol Bull. 1982;92(2):490.
43. Ghosh JK, Delampady M, Samanta T. An introduction to Bayesian analysis: theory and methods, vol. 725. Springer; 2006.