On Process Discovery Experimentation: Addressing the Need for Research Methodology in Process Discovery

JANA-REBECCA REHSE, University of Mannheim, Mannheim, Germany
SANDER J. J. LEEMANS, RWTH Aachen, Aachen, Germany
PETER FETTKE, Saarland University, Saarbrücken, Germany and German Research Center for Artificial Intelligence (DFKI), Saarbrücken, Germany
JAN MARTIJN E. M. VAN DER WERF, Utrecht University, Utrecht, the Netherlands

Process mining aims to derive insights into business processes from event logs recorded from information systems. Process discovery algorithms construct process models that describe the executed process. With the increasing availability of large-scale event logs, process discovery has shifted towards a data-oriented research discipline, aiming to design algorithms that are applicable and useful in practice. This shift has revealed a fundamental problem in process discovery research: Currently, contributions can only be considered in isolation. Researchers conduct experiments to show that they move the field forward, but due to a lack of reliability and validity, the individual contributions are hard to generalize. In this article, we argue that one reason for these problems is the lack of conventions or standards for experimental design in process discovery. Hence, we propose "process discovery engineering": a research methodology for process discovery, consisting of a shared terminology and a checklist for conducting experiments. We demonstrate its applicability by means of an example experimental evaluation of process discovery algorithms and discuss the implications of the methodology on the field. This article is not meant to be prescriptive but to invite and encourage the community to contribute to this discussion to advance the field as a whole.

CCS Concepts: • **Theory of computation** → **Design and analysis of algorithms**; • **Applied computing** → **Business process management**;

Additional Key Words and Phrases: process mining, process discovery, evaluation

ACM Reference format:
Jana-Rebecca Rehse, Sander J. J. Leemans, Peter Fettke, and Jan Martijn E. M. van der Werf. 2024. On Process Discovery Experimentation: Addressing the Need for Research Methodology in Process Discovery. *ACM Trans. Softw. Eng. Methodol.* 34, 1, Article 3 (December 2024), 29 pages.
https://doi.org/10.1145/3672447

Authors' Contact Information: Jana-Rebecca Rehse (corresponding author), University of Mannheim, Mannheim, Germany; e-mail: rehse@uni-mannheim.de; Sander J. J. Leemans, RWTH Aachen, Aachen, Germany; e-mail: s.leemans@bpm.rwth-aachen.de; Peter Fettke, Saarland University, Saarbrücken, Germany and German Research Center for Artificial Intelligence (DFKI), Saarbrücken, Germany; e-mail: peter.fettke@dfki.de; Jan Martijn E. M. van der Werf, Utrecht University, Utrecht, the Netherlands; e-mail: j.m.e.m.vanderwerf@uu.nl.

ACM 1557-7392/2024/12-ART3
https://doi.org/10.1145/3672447

1 Introduction

Process mining is a family of techniques that aims to extract knowledge on business processes from event logs, which contain records of executions of process instances [72]. One sub-field in process mining is process discovery, which analyzes an event log to construct a process model that accurately describes the executed process [75]. Since its inception more than two decades ago [16, 17], process discovery has gained widespread practical adoption, partially because tools and techniques are able to handle the complexity of real-life process data [23]. Process discovery has its roots in theoretical computer science and automata theory [75], where the main objective is to design algorithms that exhibit certain formal properties, such as producing deadlock-free models [71]. With the increasing availability of large-scale real-life event logs, process discovery has shifted towards a data-oriented research discipline, where the main goal is to design algorithms that are applicable and useful in practice [72].

Showing the applicability and usefulness of an algorithm in new settings requires a different approach. For such empirical properties, methods stemming from theoretical computing sciences are no longer sufficient. Instead researchers turn to empirical research methods to provide the necessary evidence for their respective claims. The advantage of formal properties is their "universal truth": a property is proven to always hold if its premises are fulfilled. For empirical properties, this is more complicated: it is very hard to formulate empirical properties that can be generalized to a different context. For example, the required premises for such properties are implicit or even unknown.

As an example, consider an analyst or researcher who wants to learn about the state-of-the-art on process discovery on infrequent behavior. They look into the process discovery literature to find out which algorithm is the most suitable for their discovery task. Table 1 shows four papers that provide potential solutions. Based on [7], they may infer that **split miner (SM)** "outperforms" Fodina [82]. Comparing the conclusions of HybridILPMiner [81] and SM [7], both compare to Inductive Miner Infrequent [45]. As HybridILPMiner performs comparably, and SM outperforms the Inductive Miner Infrequent in terms of F-score, they may infer that SM is the most suitable algorithm for their situation. However, there are two problems with this inference. First, research has shown that the measures on which the conclusions of these papers are based have a problem with validity [35, 36, 70]. As the premises used in the papers are violated, it is unclear how to interpret the original results of the four papers. Second, even if the results of the individual papers hold, it is questionable whether they can be compared, as a recent study showed that the results of process discovery algorithms have a problem with reliability: small variations in the event log can give totally different results [76].

This example illustrates a current problem of process discovery research. Currently, research contributions are made in isolation, as shown in Figure 1(a). In each paper, researchers conduct an experiment to show that they have moved the field forward, e.g., by showing that their newly designed algorithm outperforms the current state-of-the-art in a specific aspect. Although each experimental setup makes sense in the context of the respective paper, the above example shows that there are problems in interpreting their results because of a lack of reliability and validity in the field. As a consequence, new situations require a new experiment, as it is not possible to infer any general statements about the state of the art of the process discovery field as a whole. This generalization, or causal inference, of experimental results is a collective necessity of any research discipline: being able to infer results from different studies to draw conclusions for the field as a whole [67]. This idea is shown in Figure 1(b): experiments in individual papers provide general insights that allow for inferences to new settings. The above problems illustrate that currently causal inference in process discovery is challenging, at the least.

Table 1. Exemplary Comparison of the Experiments from Three State-of-the-Art Process Discovery Contributions

	HybridILPMiner [81]		**Fodina** [82]		**Split Miner** [7]	**Inductive Miner infrequent** [46]
Claim	HybridILPMiner is able to (1) "discover relaxed sound workflow nets" (formal property) and (2) "abstract from infrequent behavior" (empirical property)		With respect to Heuristic Miner, Fodina (1) is "more robust to noisy data," (2) is able to "discover duplicate activities" and (3) "allows for flexible configuration options" (empirical properties)		Split Miner (1) guarantees "deadlock freedom for cyclic BPMN models" (formal property) and (2) "produces simple process models, while balancing fitness, precision and generalization" (empirical property)	The Inductive Miner—infrequent (IMi) is able to "discover[.] a sound 80% model, does that fast and […] filter infrequent behaviour".
Proof	Discovery of relaxed sound workflow nets		—		Soundness of acyclic BPMN models and deadlock freedom of cyclic BPMN models	—
Experiment Setup	Two-part experiment		Two-part experiment		Comparative experiment	Comparative experiment
	Comparative experiment for ground-truth logs	Applicability test for real-life logs	Robustness measurement	Comparative experiment		
Quality dimensions	Precision, Fitness	Precision, Fitness, Simplicity	Robustness	Precision, Fitness	Precision, Fitness, Generalization, Simplicity	Precision, Fitness, Generalization, Simplicity
Quality measures	Alignment-based precision [52], alignment-based fitness [73]	Alignment-based precision [52], alignment-based fitness [73], number of arcs	Percentage of traces for which a perfectly fitting model can be discovered	Behavioral weighted precision [84], behavioral recall [29]	Alignment-based precision [3], alignment-based fitness [4], F-score, 3-fold cross-validation on F-score, model size, CFC, structuredness	Alignment-based precision [2], Cost-based fitness [4], Replay generalization [73], number of arcs, places and transitions
Contestants	Inductive Miner infrequent [46], ILP with automaton filter [15]	—	Heuristics Miner [87], Flexible Heuristics Miner [86]		Inductive Miner infrequent [46], Evolutionary Tree Miner [13], Heuristics Miner (ProM 6), Structured Miner over Heuristics Miner (ProM 6), Fodina Miner [82]	Inductive Miner [45], ILP [77], Evolutionary Tree Miner [13], Heuristics Miner [87], Flower Miner, Trace Miner
Event logs	Simulated logs [49]	Road Traffic Fines [20], SEPSIS [48]	46 synthetic logs [53], 4 real-life logs		BPIC 2012-15,17 [78], Road Traffic Fines [20], SEPSIS [48]	BPIC 2011-12 [78], WABO 1-5 (non-public)
Conclusion	The experiments show that when faced with infrequent behavior, HybridILPMiner performs comparably to [45], but is faster and less sensible to threshold variations than [15].		The experiments show that Fodina is more robust than existing heuristic miners. In addition, it is capable of mining duplicate tasks and includes various configuration options to guide the discovery.		In the experiments, Split Miner outperforms all contestants in terms of F-score (precision/fitness) and generalization. With regard to complexity, it produces comparable results. It is also notably faster than the contestants.	"The experiments show that IMi always returned a sound 80% model fast, and on all logs scores good on all quality criteria except precision."

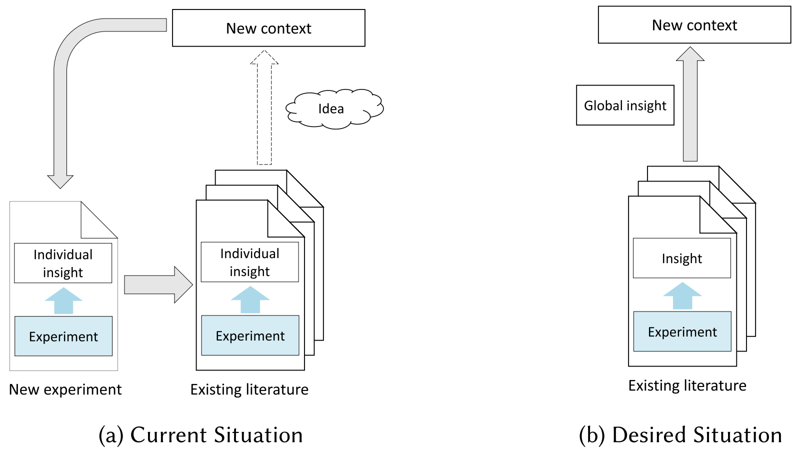

Fig. 1. Current research leads to individual insights, thus requiring to conduct new experiments in new contexts (a), whereas causal inference would lead to global insights that can be applied in new contexts (b).

We argue that one reason for the identified problems is the lack of conventions or standards for experimental design in process discovery. As we show in Table 1, researchers develop their own experimental designs, which involve many specific design decisions. This proliferation of ad-hoc research methods obfuscates the actual problem of causal inference, as many premises remain implicit. Instead of considering the reliability and validity of experiments in isolation, conventions in the form of research methodology help to (1) to design experiments more systematically, thus reducing the risks of reliability and validity issues, and (2) to explicate premises, so that we know whether or not causal inferences are possible. The goal of this article is to raise awareness of this problem in the process discovery discipline and to suggest a first step towards addressing it: *"If we understand better what we're doing, we might be able to do it better"* [24].

In this article, we advocate that a systematic approach for process discovery experiments is an essential next step towards developing process mining into a "normal science" as defined by Kuhn [42]. We argue that a commonly accepted research methodology would support researchers in their scientific inquiry by explicating the design choices and rationale behind designing and executing their experiments. To this end, we propose "**process discovery engineering (PDE)**" as a research methodology for process discovery based on **algorithm engineering (AE)** [22, 66], with a specific focus on experimentation. This methodology fulfils two purposes: it guides researchers in their own experimentation, and it structures the communication of their research results. In particular, we:

—Discuss what problems the field of process discovery has with the principles of scientific inquiry (Section 2);
—Propose AE as a suitable foundation for developing a research methodology for process discovery (Section 3);
—Develop a research methodology for process discovery experimentation that consists of a shared terminology and a checklist for conducting experiments (Section 4);
—Demonstrate the applicability of this research methodology through an example experiment (Section 5).

Rather than being prescriptive, we aim to initiate a discussion about how to perform experiments systematically and rigorously, in order to advance the field as a whole.

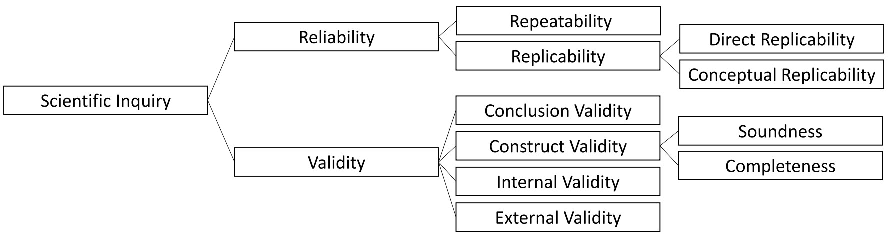

Fig. 2. Reliability and validity as basic principles of scientific inquiry.

2 Process Discovery and Scientific Inquiry

The problem of process discovery entails three major entities: process,[1] log, and model. A *(business) process P* refers to "a set of activities that are performed in coordination in an organizational and technical environment and jointly realize a business goal" [88]. Systems that support a process can capture and store records of its execution, such that they can be extracted as an *(event) log L* [13]. The goal of process discovery is to use the information from L to construct a *(process) model M* to represent P [76].

Over the last two decades, more than 100 discovery algorithms have been proposed [69, 72]. Initial algorithms, such as alpha miner [75], ILP miner [77] and Inductive Miner [45] come with formal guarantees, but without an experimental evaluation. Table 1 shows four exemplary papers that both introduce and experimentally evaluate discovery algorithms. These algorithms are well-cited, widely accepted and frequently used in the process discovery community. They cover different families of discovery algorithms developed by different research groups over the last ten years.

As exemplified in Table 1, most authors follow a similar approach for their evaluation: they choose one or multiple event logs and apply an implementation of their algorithm to these logs. Sometimes, other algorithms are also applied to the same logs for comparison. From the obtained results, propositions are inferred about the algorithms and their quality. Once published, the community uses these propositions in decision-making: practitioners decide whether the algorithm is applicable in their context, and scientists decide which problems to address in their own research. Such decision-making is an example of causal inference, i.e., extending a causal relationship across or beyond the studied conditions [67]. However, whether and under which circumstances such an inference of published evaluations is justified remains an open question.

Studying such questions is meta-research [24]: a critical evaluation of how research is conducted and inferences are made. In essence, most qualities meant to evaluate research can be reduced to *reliability* and *validity* [68]. Reliability relates to the consistency and stability of the results, whereas validity concerns the integrity of the conclusions and the absence of random and systematic errors [68]. In the remainder of this section, we apply these principles to the field of process discovery, as illustrated in Figure 2. We consider *experimentation* to be the process of conducting experiments. *Results* are the outcome of experimentation, i.e., inferences based on experimentation.

2.1 Reliability

The first principle, *reliability*, is the degree to which a measure of a concept is stable [68]. Reliability considers both *repeatability*, i.e., whether the results are stable over time, and *internal reliability*, i.e., whether the measures are consistent and replicable. Repeatability is often referred to as test-retest reliability or stability and expresses to what degree the same researchers obtain similar results

[1]This entity is sometimes called "system" [13] but we consider in our context the term "process" to be more tangible [36, 76].

Table 2. Exemplary Analysis of Open and Mitigated Threats to Reliability

		Repeatability	Direct replicability	Conceptual replicability
HybridILP [81]	Mitig.	Event logs are randomly modified, but only once, i.e., for each alpha, there is only one log	Algorithm implementation and event logs are made available	No evidence against it
	Open	Algorithm is non-deterministic, which is not accounted for	Parameters of contestant algorithms are not provided	No evidence for it
Fodina [82]	Mitig.	Event logs are constructed in a randomized way, but the evaluation always relies on the same logs	Algorithm implementation and experimental logs are made available	No evidence against it
	Open	HM is not deterministic in nature, so the authors might encounter different results when repeating the experiment	Conversion procedure of HM is not made available	No evidence for it
SM [7]	Mitig.	No random elements, so repeatability is given. Execution times are repeated five times to counteract random effects	Algorithm implementation is made available, only public logs are used	No evidence against it
	Open	None	Optimized parameters for contestants are not provided	No evidence for it
IMi [46]	Mitig.	There does not seem to be any random elements to the evaluation, so repeatability is given	Algorithm implementation is made available	No evidence against it
	Open	None	The authors do not specify how the "80% models" are achieved for the contesting miners	No evidence for it

when they redo the experiment. Replicability,[2] also called internal reliability [68], is the ultimate standard by which scientific claims are judged [56]. Following [31], we distinguish two types of replicability: *direct replicability* and *conceptual replicability*. An experiment is directly replicable if independent researchers can repeat the experiment with the same experiment setup on the same data and obtain similar results. Conceptual replicability of an experiment refers to whether similar results are obtained when conducted on other data or with a different setup: "this type of replication is often linked by commentators to the goal of generalizing a finding or of testing its robustness, rather than assessing its reliability" [31].

Table 2 illustrates the principles of reliability, with examples of potential threats to reliability for each of the four discovery papers of Table 1. Some of the potential threats have been mentioned in the respective papers: they were explicated and the experimental setup was adapted to mitigate them. Other potential threats were not explicated in the papers and hence remain open: we do not know whether they were accounted for in the experimental setup, or whether they affect the results of the experiments.

Repeatability. In process discovery, most experiments are conducted in an automated way, which ensures repeatability. However, threats to repeatability can occur e.g., when the experiment includes randomization or when the algorithms contain non-deterministic computations. As shown in Table 2, some experiments rely on randomly modified event logs but use the same logs for all experiments, such that the randomization does not interfere with the results. Experiments that rely on algorithms that are non-deterministic by nature should account for this in their experimental setup. For example, ILP miner [77] relies on solving a set of ILP problems such that results of one step influence the ILP problem in the next step. When algorithms are used in an experiment that relies on such non-deterministic steps, e.g., [81, 82], stability of the results is not ensured, and thus

[2]Also referred to as reproducibility; although replicability and reproducibility are closely related, there is no consensus on their definition [31].

a threat the experimental setup needs to address is to study whether the reported results are stable when repeated, and not "just" a single outlier.

Direct Replicability. In process discovery, direct replicability has always been a core value, witnessed by publicly available implementations of many process discovery algorithms in open source frameworks (e.g., [10, 79]), as well as publicly available event logs[3] and evaluation frameworks [6]. As Table 2 shows, all papers have made their algorithms publicly available. However, it is not always possible to replicate the experiments from the described experimental setup, as, for example, most papers do not explicate the exact parameter settings of the algorithms used in the experiment. As a consequence, it cannot be checked whether obtaining the reported results requires optimizing parameter settings or whether the default settings suffice.

Conceptual Replicability. In process discovery, conceptual replicability directly addresses real-life challenges: given a new, non-studied event log, will the application of the algorithm be consistent with previously obtained results? Achieving conceptual replicability for process discovery is an inherently difficult problem to address, as any experimentation using a limited set of event logs typically disallows broader conclusions beyond the boundaries of the used data [59, p49]. Indeed, current experimentations with process discovery algorithms only assess the applicability, feasibility and performance of the algorithm (see Table 1). Although necessary, these experimentations do not achieve conceptual replicability [57, 76].

None of the algorithms presented in Table 1 provides evidence whether and to what extent the experimental results can be transferred to a new context, i.e., how the algorithm would behave on other event logs. For example, two papers [81, 82] use event logs generated from known models but do not compare the outcomes of the experiment with the original models. Similarly, the other two papers [7, 45] only use real event logs, for which the underlying processes are unknown. This means that the relation between input and output data is not studied. Thus, it is unclear whether the results are caused by properties of the original model or by the algorithm. As such, there is no evidence *for* conceptual replicability. On the other hand, these papers also do not provide any evidence *against* conceptual replicability, i.e., design choices which suggest that the algorithms cannot be applied in a different setting. For example, all papers use several different logs in their experiments, for which the results are consistent.

One way to provide evidence for conceptual replicability would be to establish a relation between the input (e.g., properties of the event log, such as size, variability, etc.) and the output (e.g., the quality of the discovered model) [76]. For example, a paper could experimentally show that when we add up to 5% infrequent behavior to the log, the model quality does not change significantly. This could be achieved in a controlled experiment with synthetically generated models and logs by adding infrequent behavior in 1% increments and statistically assessing the changes in model quality. Although this sounds like a simple task, it is far from trivial: A recent study [76] has shown that small variations in the input event log can significantly impact the quality of the resulting process model, both positively and negatively, i.e., that such a relation has not yet been established. As a consequence, the best discovery algorithm for a given event log is typically determined empirically on a case-by-case basis [62].

2.2 Validity

If results are reliable, they are not necessarily also valid [68]: "validity describes whether our operationalizations and collected data share the true meaning of the constructs we set out to measure" [59]. In other words, the research propositions should be free from random and systematic

[3]https://www.tf-pm.org/

errors [68]. Different types of validity have been identified, of which *conclusion validity*, *construct validity*, *internal validity*, and *external validity* are the most frequently used [67].

Conclusion validity refers to the validity of inferences about the correlation between treatment and outcome [67]. Internal validity means that effects observed in an experiment can be causally attributed to the applied treatment [67]. Construct validity refers to the validity of inferences about the constructs that represent an experimental treatment and outcome [67]. Here, construct denotes a measurable operationalization of a real-world phenomenon [59]. Finally, external validity concerns the generalizability of the results. It considers to what extent we can assume the results of an experiment to hold "beyond the sample or domain that the researcher observes" [44].

To explain the types of validity, consider Table 3. It provides examples of potential threats to validity for the four papers in Table 1. Similar to Table 2, some of the potential threats to validity have been explicated and mitigated, while others were not explicated and thus remain open.

Conclusion Validity. In process discovery, conclusion validity relates to the claimed properties of an algorithm. It concerns the extent to which the experimentation results allow for the conclusion of these properties [51]. One such property, which is frequently claimed by new discovery algorithms is that they outperform existing contesting algorithms in terms of model quality. This property is typically assessed by applying the algorithms on the same set of event logs and comparing their quality measurements. However, such a comparison can be unfair towards the contesting algorithms if they are not allowed to perform at their best ability, e.g., if their parameters are not optimized for the given setting. This threat can be mitigated by optimizing the parameters of the contesting algorithms individually (see Table 3). Other potential threats to conclusion validity are a mismatch between claim and experiment (i.e., a paper might claim an algorithmic property, which is not or not completely shown) or a lack of experimental data (i.e., the paper does not report on all experimental results, meaning that nothing can be concluded from the experiment).

Construct Validity. In process discovery, construct validity determines "whether a measure of a construct sufficiently measures the intended property" [51]. Because process discovery constructs typically relate to a system of formal definitions, theorems and algorithms, we argue that *soundness* and *completeness* are necessary prerequisites [28] for construct validity. Shadish et al. [67] list the inadequate explication of constructs and the incompleteness of construct definitions (referred to as "construct confoundness") as two major threats to construct validity. Hence, an important aspect in addressing construct validity is to ensure the adequate explication of constructs and approximation of measures [67].

The most obvious constructs that we deal with in process discovery are event logs, process models and quality measures. However, as the examples in Table 3 show, these constructs are sometimes neither sufficiently well defined nor operationalized, and thus a potential threat to validity. Additionally, Table 1 shows no indiviual measure is used in all four papers. In other words, it is hard to compare the results between the papers. These observations are part of a larger discussion on the properties of quality measures, like precision, generalization and F-score. Compared to other disciplines, process discovery cannot rely on uncontested measures to judge the quality of its results. Research has shown that these measures often do not measure what they intent to measure (cf. [5, 35, 57, 70]). For example, precision is not well-defined as a construct [70], so it is challenging to measure [35]. Another problematic measure is the F-score, which is often used to capture the quality of models because it claims to balance fitness and precision, but "there are substantial negative correlations between fitness metrics and precision metrics. As such, models with a good fitness typically have a low precision, and vice versa" [35], implying that such a balance does not exist, or at least requires a better definition [90].

Internal Validity. In process discovery, internal validity entails whether the observed properties of a discovered model can be attributed to the applied algorithm, as "a research design is internally

Table 3. Exemplary Analysis of Open and Mitigated Threats to Validity

		Conclusion validity	Construct validity		Internal validity	External validity
			Soundness	Completeness		
HybridILP [81]	Mitig.	All evaluation data is provided as supplemental material	“Infrequent behaviour” is operationalized as a filtering function, whose exact definition is left to the user	The algorithm is completely specified	The modifications of the event logs are explicated and justified	The algorithm is compared against both the original ILP, which is to be improved, and a state-of-the-art algorithm (IMi)
	Open	The authors conclude that their approach delivers “more expected” results but do not explicate those expectations	Precision measure is imprecise (problem was not known/ published at the time [70])	“Exceptional” behavior (non-compliant) is not distinguished from “infrequent” behavior (compliant, but rare)	It is unknown how much infrequent behavior the event logs originally contained, so this effect might be observed in the experiment	The authors advise to use lower threshold values to yield less complex models, however, this is not grounded in the experiment
Fodina [82]	Mitig.	Contestant parameters are justified (most relaxed parameters to ensure fairness)	The authors conduct and justify a “robustness check,” which is a fitness evaluation adapted to the representational bias of a modelling language	Process models are defined as Causal Nets, all returned models adhere to this definition	All evaluation data are presented, if measures are missing, the reason is provided	Selection of event logs is justified and fits the purpose of the evaluation (robustness check)
	Open	The claim they make is not the one they evaluate (absolute vs relative)	F1-score is used to compare discovery algorithms (problem was not known/published at this time [35])	“Noise” and “robustness” (as a derivative) are never defined	The modified conversion procedure in case of HM is not explicated or justified	The evaluation is only relative to HM and FHM, which were not state-of-the-art, even at the time of publication
SM [7]	Mitig.	Parameters are optimized for all contestants	All measures are defined and explicated	An event log is defined conceptually and formally. All logs adhere to this definition (XES)	The experimental design is comprehensively explicated	All state-of-the-art miners were chosen as contestants; a large collection of available real-life logs is used
	Open	Many values are missing in the results table (likely due to unsound models), but the potential impacts are not discussed	F1 score is used to compare discovery algorithms (problem was published at the time [35])	The overall objective of the experiment is not stated	Models are translated between Petri nets and BPMN to be measured, which might impact the results	Some logs are prefiltered before discovery, which impacts the results of all miners. This means we cannot conclude anything about logs with infrequent behavior
IMi [46]	Mitig.	The authors take multiple actions to ensure measurability of contesting results	All resulting models are converted into Petri Nets and compared on that basis	The experiments relate to all claims of the paper (fast, sound, higher quality)	All data is reported on (including the reasons for missing values)	All state-of-the-art miners (of that time) were chosen as contestants
	Open	Many values are missing in the table (likely due to missing alignment computations), but the potential impacts are not discussed	The time required for model conversions is included in the overall computation time, so some miners are put at a disadvantage	The concept of an “80% model” is not specified or defined	HM and ETM do not produce process trees, so the results are translated. Potential errors in the conversion are not accounted for	All miners (including IM and IMi) appear to not perform well on larger logs (e.g., BPI’11 and BPI’12). This is not discussed anywhere

valid when its manipulation is causally responsible for an observed effect” [51]. As shown in Table 3, multiple pre- and postprocessing steps can be applied to the event logs before and after discovery. These steps can include modifications to the event log, such as filtering out traces and/or event types, or modifications to the process models, such as conversions between different formalisms. As these steps can influence the results both positively and negatively, their effects must be accounted for when evaluating the results.

Another important example of a threat to internal validity in process discovery experimentation is sampling, which may have a large influence on the discovered models [40]: sampling may skew or invalidate conclusions. On the one hand, if a sample is taken from a well-structured event log, i.e., a log with many frequent and only very few infrequent traces, sampling prior to discovery will yield a comparable model quality with a lower runtime [8, 25]. If an experiment only contains such logs, a positive correlation can be found between sample size and model quality. On the other hand, for logs containing mostly infrequent traces, sampling has a negative influence on the quality of the discovered models. Unfortunately, there are currently no proper measures to evaluate or compare samples of event logs [76]. Consequently, the properties of the event logs selected for an evaluation can have a major influence on the observed results and should therefore be considered in designing the experiment.

External Validity. In process discovery, external validity amounts to the capability of an algorithm to discover the "true process" for any given event log. The external validity of the discovered model then relates to the degree to which it reproduces this ground truth. However, this is mainly a theoretical consideration because this ideal setting may not always be possible [30]. Even if it is, it is hardly achievable in practice. Such a ground truth only exists in a lab setting with synthetically generated process models, assuming there is a one-to-one correspondence between process and model [37, 76]). One example is the recently started Process Discovery Contest, where the results of discovery algorithms on a given set of event logs are compared with the ground truth model from which the event logs where generated. In such cases, the degree of reproduction can be assessed by means of model similarity measures, such as trace equivalence or bisimulation, but they suffer from representational bias, among other issues, and only work for models with a formal foundation, which severely limits their applicability.

In field settings using real-world event logs, the true process is usually unknown. Therefore, external validity has to be assessed in a different way. As shown in Table 3, the "ground truth" model is approximated by a model that scores high in the quality dimensions that the algorithm targets (e.g., fitness and precision). Even though these quality dimensions provide some level of confidence in the algorithm itself, the above-mentioned problems (see the discussion on conclusion validity) imply that tradeoffs between these quality dimensions have to be made, either by the algorithm itself or by the analyst. However, it is unknown whether these tradeoffs bring you closer or further from the ground truth model. Besides, it is not clear how the quality dimensions relate to the ground truth model. Research has shown that the quality dimensions assess the relation between model and log, not between process and log [35, 36, 61, 76], i.e., there currently is no evidence whether the discovered model resembles the process well.

Random sampling simplifies external validity inferences because it eliminates possible interactions between the causal relationship and the studied samples [67]. However, this is rarely feasible in process discovery evaluations. Instead, as shown in Table 3, researchers often evaluate their process discovery algorithms on specific sets of real-life event logs, assuming that these logs cover at least parts of the variability that can be expected in other logs [51]. However, it is unknown whether this is a valid assumption. Therefore, evaluating on real-life logs does not allow for general propositions about external validity [51, 59]. To mitigate this issue, researchers have only recently started to study what needs to be done to ensure external validity in process discovery [35, 38, 76].

3 Process Discovery and Research Methodology

One way to ensure reliability and validity is to apply rigorous research methods. Developing such methods is the focus of the field of research methodology, i.e., the study of methods [26]. Stemming from theoretical computer science, early process discovery did not have the need for explicit and agreed-upon research methods. However, the shift towards a practical, data-driven discipline warrants such methods [42]. In this section, we explicate the research methods of the process discovery field and demonstrate that the field already follows a research methodology, though implicitly. The explication allows us to reflect on our research methods to address the reliability and validity problems discussed in the previous section.

3.1 PDE

Methodology describes strategies of inquiry in a certain field [59]. Strategies of inquiry are types of research method designs that provide specific direction for procedures in a research design [19]. In other research fields, these include empirical strategies [59], such as quantitative and qualitative methods. However, for algorithmic research this is less straightforward.

One strategy of inquiry that is commonly used for the design of artifacts in Information Systems research is Design Science [33, 55, 89]. However, Design Science Research provides mainly methods and techniques to study artifacts in context. For example, Hevner's three-cycle framework [32] introduces the relevance cycle to study how an artifact can be applied in a given socio-technical environment to solve a concrete problem. Similarly, the rigor cycle focuses on the "selection and application of appropriate theories and methods for constructing and evaluating the artifact" in the socio-technical environment [32]. Although process discovery algorithms can be seen as artifacts, an algorithm should work in any context, provided that its preconditions are met [18]. Thus, algorithm designers want to make claims about an algorithm that is *independent* from the context of its use.

A research methodology that studies algorithms independent of their context is AE [66], which encompasses both theoretical and empirical strategies. It is the discipline concerned with the design, analysis, implementation, tuning, debugging and experimental evaluation of computer programs for solving algorithmic problems [66]. AE integrates algorithm theory with the principles of experimental validation coined in Popper's scientific method [58]. Instead of considering algorithm design and analysis on the one hand and implementation and experimentation on the other hand, AE prescribes a feedback loop of design, analysis, implementation and experimentation that leads to new design ideas [66].

AE provides a research cycle subdivided into two parts: *Design* and *analysis* constitute the formal part of AE, which corresponds to the theory building in Popper‘s scientific method [58, 66]. *Implementation* and *experimentation* constitute the empirical part of AE, which corresponds to the experimental validation in Popper's scientific method [58, 66]. In process discovery, the same research activities can be identified. Therefore, we apply AE to process discovery, resulting in PDE, depicted in Figure 3. PDE is the discipline concerned with the design and analysis (coloured orange in Figure 3) as well as with implementation and experimentation (coloured blue in Figure 3) of process discovery algorithms to study properties of these algorithms independent of their context.

3.2 Explicating PDE in Process Discovery Research

Process discovery has its roots in theoretical computer science, which focuses on the design and analysis of algorithms, the first two activities in PDE. For example, the main goal of the first process discovery algorithm, α-miner, was to theoretically show that under certain conditions, the original model that generated the event log is rediscoverable, i.e., that the algorithm returns an isomorphic model [75]. As such, the field has a strong background in showing formal properties of process discovery algorithms, which in PDE are part of the activities of design and analysis.

As shown in Table 4, the included papers implicitly follow the line of reasoning from PDE: they motivate the need of a new algorithm by generalizing observations from realistic contexts. For example, HybridILPMiner [81] observes that many realistic contexts have to deal with infrequent behavior and takes this a starting point to improve ILP Miner [77]. SM [7] was developed from a

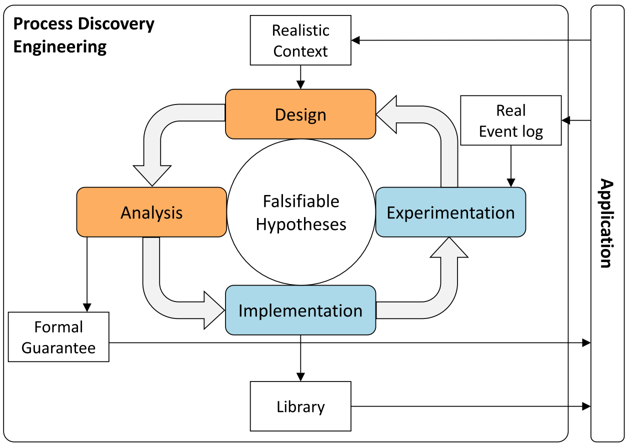

Fig. 3. PDE: AE [66] applied to the field of process discovery.

different observation from realistic contexts: that block-structured models over-generalize behavior. Based on these observations, the authors then design algorithms that address the identified problems.

Algorithms can be extensions of existing discovery techniques or be completely new approaches to process discovery. The algorithm designs are typically presented formally, in the form of mathematical formulas and pseudo-code, which enables the analysis of these algorithms and provides formal guarantees by means of proofs. These guarantees may range from properties of the results of the algorithm to properties of the algorithm itself. For example, the SM algorithm is accompanied by two formal guarantees: its worst-case runtime complexity and deadlock-freedom of the resulting models (see Table 4). Table 4 also shows that there is some variety to how researchers approach the analysis step, and whether it is included at all. In some cases, the analysis is separated from the experimentation (see below), and the results are published separately.

With the growing maturity of discovery techniques and the increased availability of event log data (real event logs), process mining has shifted from a formal to a more practical data-driven discipline [72]. This means that the field has developed a strong emphasis on the implementation of and the experimentation with algorithms. As Table 4 exemplifies, nearly all discovery algorithms come with an implementation in one or more frameworks, such as ProM [79] or PM4Py [10]. Through experimentation, the authors show the feasibility of their newly designed algorithms, as elaborated in Table 1. As the examples in Table 4 show, process discovery research follows the activities in PDE, though this is typically not explicated.

3.3 Current Problems in Experimentation

As shown in the previous section, process discovery research already implicitly follows PDE. The first three activities, design, analysis and implementation, are established from the field of theoretical computer science and engineering. However, for experimentation, we have shown in Section 2 that the existing research methods do not sufficiently ensure reliability and validity, particularly with regard to conceptual replicability and external validity. In the following, we review the main research methods currently used for process discovery experimentation and discuss their shortcomings.

Case Studies. In process discovery, case studies aim to obtain knowledge on e.g., applicability, studying "contemporary phenomena" (business processes) in their "natural context" (organizations) [65]. To help guide the planning and execution of single-case experimental designs, multiple

Table 4. Exemplary Comparison of the Experiments from Three State-of-the-Art Process Discovery Contributions

	HybridILPMiner [81]	**Fodina** [82]	SM [7]	**Inductive Miner infrequent** [46]
Realistic Context	Section 1: Because of low-frequent behavior, ILP-based process discovery often yields unsatisfactory results.	Section 1: Heuristic miners have multiple issues that negatively impact their reliability.	Section 1: Tradeoffs between four quality dimensions have proved elusive; block-structured models overgeneralize behavior in the event log.	Section 1: No process discovery technique can discover a sound 80% model fast and filter infrequent behavior.
Design	Section 4: Discovery of relaxed sound workflow nets, Section 5: Extend discovery algorithm by a pruning approach to handle low-frequent behavior	Section 4: Robust and flexible discovery algorithm based on the heuristic miner	Section 3: SM algorithm	Section 3: Extension of the original inductive miner with filtering capabilities for infrequent behavior
Analysis	Section 4: Semantic properties of discovered model (Theorem 2)	None	Section 3.8: Complexity analysis, Section 4: Semantic properties of discovered model (Theorem 4.8)	Proof of properties of IM in a different paper [45]
Formal Guarantees	Discovered models are relaxed sound	None	SM has complexity $O(E + A^6)$, where E is the number of events in the log and A is the number of activities, Soundness of acyclic BPMN models and deadlock freedom of cyclic BPMN models	IMi returns block-structured models (thus sound)
Implementation	Section 6: Code provided via GitHub & SVN	Section 4.1: Code provided via website	Section 5: Code provided via website	Section 1: Code provided through library
Libraries	ProM, RapidProM	ProM	Apromore	ProM
Falsifiable Hypotheses	Nonexistent (only claim)	Nonexistent (only claim)	Nonexistent (only claim)	Nonexistent (only claim)
Experimentation	Section 6: See Table 1	Section 5: See Table 1	Section 5: See Table 1	Section 4: See Table 1
Real Event Logs	Simulated logs [49], Road Traffic Fines [20], SEPSIS [48]	46 synthetic logs [53], 4 real-life logs	BPIC 2012-15,17 [78], Road Traffic Fines [20], SEPSIS [48]	BPIC 2011-12 [78], WABO 1-5 (non-public)

authors suggested methods for process mining projects [11, 72, 80]. However, these methods focus on properties of the results of the algorithms, rather than on properties of the algorithms themselves. Following a method does not ensure external validity [23]: if two process discovery algorithms are shown to be applicable in a real-world setting, but the respective contexts are fundamentally different from one another, no knowledge about the relation between the two algorithms can be obtained [41]. Even though such case studies provide valuable information about the feasibility of the algorithm, their results do not allow to infer conclusions about the field as a whole.

Comparative Experiments. In process discovery, *comparative experiments* demonstrate how a new algorithm advances the state-of-the-art with regard to a specific aspect or quality measure (cf. [81, 82]). The algorithms can be applied "in vitro" to synthetic logs generated from known ground truth models [37] or "in vivo" to an available set of event logs, for which no ground truth model exists [6]. As we have shown in Table 1, different comparative experiments employ different experimental setups, which leads to incomparable results [6, 76]. To ensure the validity of such a comparison, there is a need for a commonly accepted and scientifically valid framework for the systematic evaluation of different process discovery approaches and measures [36]. Multiple attempts have been made to provide such a framework [63, 64, 83, 85], but these frameworks do not provide the guidance, level of detail or adaptability necessary to ensure reliability and validity of process discovery experiments [23, 36, 57, 60, 76]. Hence, it cannot be established how the outcomes of such experiments contribute to the field as a whole.

Benchmarks. In process discovery, *benchmarks* provide standardized settings to compare different algorithms experimentally. The first benchmark evaluation of discovery algorithms concluded that particularly complex event logs were still challenging for the state-of-the-art algorithms of 2012 [21]. The second one, 7 years later, concluded that algorithms had become more mature but still struggled to find appropriate tradeoffs between quality dimensions [6]. From a methodical viewpoint, a benchmark is a first step towards increased reliability as it examines algorithms in a common setting. Although competitive testing provides knowledge about which algorithm is better with respect to specific accuracy metrics, it provides limited understanding of why [34] and does not solve the problems of validity. It shifts the responsibility of ensuring reliability and validity of an experiment from the individual paper to the benchmark method, but it does not address the discussed issues themselves. For example, the discussion on the validity of measures is independent of how the evaluation is performed. Although benchmarks are a valuable addition to the field, they do not solve the discussed problems.

As this discussion shows, the current problems of reliability and validity mainly stem from the inadequacies of the currently available research methods for experimentation in process discovery. In the next section, we argue that for empirical process discovery, each individual research work needs to proactively establish its own claims in such a way that it can be understood how they contribute to the field as a whole. The field of process discovery requires new methods to achieve that goal.

4 A Methodology for Experimentation

In this section, we provide methodical support for the fourth phase of PDE: experimentation. Any evaluation that is not an analysis of the algorithm is an experiment and should thus be critically assessed on the reliability and validity of its results and inferences. Consequently, most of the experiments in process discovery can be classified as quasi-experiments [67]. As for any valid and reliable experiment, causal inferences from any quasi experiment must meet the basic requirements for all causal relationships: that cause precedes effect, that cause co-varies with effect, and that alternative explanations for the causal relationship are implausible [67]. This requires a thorough design and assessment of the experiment as a research method in process discovery. Following [26], a research method consists of (i) a terminology including terms to characterize the underlying ontological and epistemological position and generic notions of research and (ii) a corresponding process. In this section, we propose a terminology for research methods in PDE in Section 4.1 and provide a checklist to assist in ensuring the quality of the used research method in Section 4.2.

4.1 A Shared Terminology

A common terminology already exists for the foundational entities of process discovery, such as log, model and process [14]. However, this terminology is not sufficient to argue about research methods, which refers to generic notions of research, such as “hypothesis” or “experiment.” To serve as a foundation for PDE, we extend the terminology, as shown in Figure 4, which consists of three parts: formal (in orange), empirical (in blue), and generic notions of research (in white).

Setting the Context. In process discovery, a Process [88] refers to the unknown entity that is the ultimate study subject of process mining projects. A Model describes one or more processes. An Event Log is a dataset describing the execution of a process. An Algorithm is a well-defined sequence of computational steps that transforms some input into some output [18]. An Algorithm implementation is an instantiation of the algorithm in a specific library [66]. For PDE, event logs are (part of) the input, and discovered models (part of) the output. Because the exact definitions

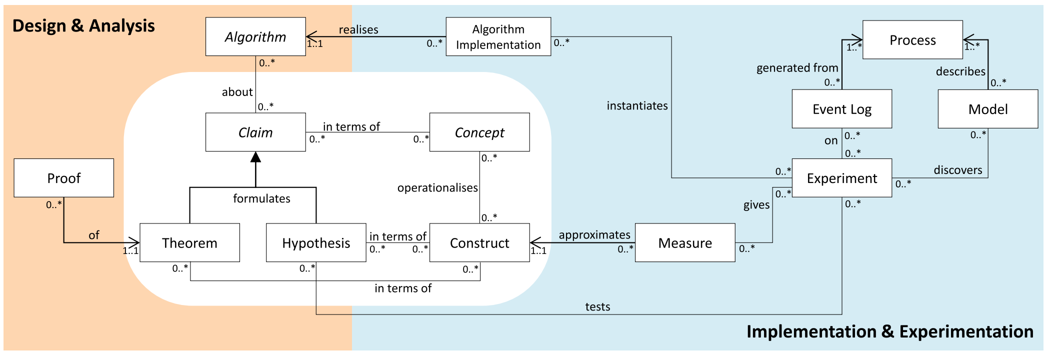

Fig. 4. Terminology for process discovery experimentation.

of models and event logs depend on the design of the discovery algorithm, we leave them to the individual authors.

Building the Hypotheses. A Claim is an alleged proposition about one or more algorithms. Validating a claim contributes to scientific knowledge. The objective of scientific experimentation is to provide a sufficient amount of evidence for validating a claim. A Theorem is a type of claim that can be validated by a formal Proof, resulting in a formal guarantee. A Hypothesis is a type of claim that cannot be validated by a formal proof, but is formulated in such a way that it can be tested statistically or empirically in an experiment [58].

A Concept "describes an abstract idea that is inferred or derived from instances that we perceive in the real world, that is, mental representations that we develop, typically based on experience" [59]. Concepts provide a mechanism to describe a claim in natural language: a concept gives an intuitive understanding of an idea, rather than a mathematical definition. To evaluate the claims, these concepts have to be specified. A Construct "is an operationalization of a concept in such a way that we can define it by measuring the construct against data" [59]. The process of operationalizing concepts into constructs allows to formulate claims as theorems or hypotheses. Therefore, constructs should be specified as precisely as possible [43, 59]. Ideally, this is achieved by providing a mathematical definition for each construct. However, many concepts cannot be specified in terms of fully formal constructs.

The terms *concept* and *construct* often cause confusion in the field of process mining. We often use the word concept (e.g., in conceptual modelling), whereas in research terminology, these are actually constructs, as the conceptual model operationalizes the abstract ideas of the concepts. Concepts and constructs can have complex interrelations if, for example, the same concept is operationalized by multiple constructs or vice versa.

Conducting the Experiment. If building a claim results in a testable hypothesis, an experiment needs to be conducted that allows to test the hypotheses. An Experiment is a research design to study a deliberate action followed by systematic observations of what occurred [67]. For example, the execution of a process discovery algorithm is a deliberate action whose effects can be observed and studied in an experiment. Experiments range from controlled experiments to field studies, including case studies, comparative experiments, and benchmarks as described in Section 3.

A Measure is a function that assigns a non-negative real value to an input. In our context, this input can be a log, a model, an action, or a combination thereof. Conceptually, measures are used to observe these effects: we consider a measure to be an empirical indicator that allows us to approximate the underlying constructs [59]. The experiment design should include a specification of how the measures are obtained and analyzed to draw conclusions about the hypotheses, for example, by explicating which statistical tests will be used. Executing the experiment results in actual values for each of the defined measures. Those values need to be further analyzed to either reject or accept the hypotheses. To ensure the (construct) validity of measures, they need to be properly operationalized in the experiment design, which is far from a trivial task. For a thorough discussion on this matter, we refer the reader to [27].

4.2 A Checklist for Experiments

The purpose of research methods is to assist the researcher in obtaining reliable and valid results. As shown in Section 3, a plethora of research methods is used for process discovery experimentation. It is impossible to exhaustively list all methods that ensure valid and reliable results. Instead, we propose a checklist for the experimentation phase of PDE, shown in Figure 5. This checklist is intended to help researchers to validate their research method against the principles of scientific inquiry and justify why executing their research will result in reliable and valid conclusions for the field as a whole. Following the checklist contributes to transparency of the experiments and the results and therefore helps to increase the validity and reliability of the results.

When conducting an experiment, there are three major steps to take: (1) devise one or more hypotheses to be validated, (2) design and execute a method that is capable of validating or falsifying these hypotheses, (3) discuss the implications and conclusions that follow from the results. Following this distinction, the checklist is divided into three major parts. Each part contains those elements (entities and relations) from Figure 4 that are relevant to this step in the experiment. Note that two elements (the entity claim and the relation between hypothesis and experiment) are repeated in the checklist, as these are relevant to two steps in the experiment: the claim should be stated in the beginning to motivate the experiment and then discussed again at the end, after its validity has or has not been established. Similarly, the hypotheses require a careful experiment design to achieve sufficient reliability and validity, and, once the results are obtained, a scrupulous examination to establish whether and how conclusions can be drawn, reviewing the principles of scientific inquiry. As this article is focused on the empirical part of PDE, we only consider the elements concerning generic notions of research (in white) and the empirical part (in blue).

For each element, the checklist provides questions to assess the research against the principles of scientific inquiry, as discussed in Section 2. Each question is annotated with the corresponding principle of scientific inquiry. To derive these questions, we assessed how each element may impact the principles of scientific enquiry (cf. Figure 2), and how and to what extent PDE researchers can ensure these principles in their experimentation. As a result, the generality of the questions varies. Some questions are concrete and easy to verify, such as “Is the construct ‘event log’ well-defined?”. Others are more generic, such as “Is the rationale for the experiment design explicated?”. Such questions are more difficult to check, as they require a justifying rationale. The varying degree of concreteness provides researchers the freedom to design their research method based on the requirements and context of their experimentation.

The structure of the checklist is generic: It needs to be adapted to the context of each individual experiment. For example, parameters or algorithm implementations can become part of the hypothesis building when studying an algorithm and the effects of its parameters. Also, the checklist is not exhaustive: each research method is unique. Instead, the generic nature of the questions in the checklist allows researchers to reflect on how the principles of scientific inquiry influence their experimentation and how they can be used to conduct high-quality research.

5 An Example Experimentation

In this section, we illustrate the applicability of the checklist. The checklist in itself does not guarantee validity and reliability of an experiment, which need to be established through review.

1. Hypotheses Building

Claim

- ☐ Are all claims related to the experimentation explicated? (Completeness)
- ☐ Are all these claims used consistently? (Soundness)

Concept

- ☐ Are all relevant concepts explicated? (Completeness)

Claim ⇆ Concept

- ☐ Do the claims use concepts correctly? (Soundness)
- ☐ Do the claims use only explicated concepts? (Completeness)

Construct

- ☐ Are all constructs sufficiently specific? (Soundness)

Concept ⇆ Construct

- ☐ Are all concepts operationalised as constructs? (Completeness)
- ☐ Is there a rationale for each operationalisation? (Soundness)
- ☐ Are all assumptions in the operationalisations explicit and applicable? (soundness)

Hypothesis

- ☐ Are all evaluated hypotheses explicated? (Completeness)
- ☐ Are all evaluated hypotheses testable? (Soundness)

Hypothesis ⇆ Claim

- ☐ Are all claims operationalised into hypotheses? (Completeness)
- ☐ Is there a rationale for the operationalisation? (Soundness)
- ☐ Are all assumptions in the operationalisation explicit and applicable? (internal validity)
- ☐ Can the claims be substantiated using the hypotheses? (conclusion validity)

Hypothesis ⇆ Construct

- ☐ Are all constructs in the hypotheses explicated? (Completeness)
- ☐ Do the hypotheses use constructs correctly? (Soundness)

Claim ⇆ Algorithm

- ☐ Is the relation between claims and algorithm(s) explicated and, if necessary, justified?

2. Method Design and Execution

Experiment

- ☐ Is the experiment design described such that the experiment can be repeated? (Direct and Conceptual replicability)
- ☐ Is the rationale for the experiment design explicated? (Internal validity)
- ☐ Are the effects of missing experiment data discussed? (Conclusion validity)
- ☐ Are other potential threats to reliability and validity described and mitigated if possible?

Hypothesis ⇆ Experiment

- ☐ Does the experiments' design allow to test the hypotheses? (Internal validity)

Algorithm implementation

- ☐ Is there a rationale for the algorithm implementation selection? (Internal validity)
- ☐ Is the implementation publicly available, or is the replicability of the implementation ensured? (Direct replicability)

Algorithm ⇆ Algorithm implementation

- ☐ Is the implementation of the algorithm correct? (Conclusion validity)

Algorithm implementation ⇆ Experiment

- ☐ Is the computational environment specified? (Repeatability)
- ☐ Are the implications of the implementation on the hypotheses addressed? (Internal validity)
- ☐ Are the parameters for the algorithms described? (Repeatability)
- ☐ Is there a rationale for these parameters? (Conclusion validity)

Process

- ☐ Is the concept "process" explicated? (Completeness)
- ☐ Is there a rationale for the selection of processes? (External validity)

Event Log

- ☐ Is the construct "event log" well-defined? (Completeness)
- ☐ Do all event logs adhere to their definition? (Soundness)
- ☐ Is there a rationale for the selection of event logs, and a discussion of its implications? (External validity)

Event Log ⇆ Process

- ☐ Is there a rationale on how the traces were collected from the process, and is this way suitable to validate the hypotheses? (Construct validity)

Experiment ⇆ Event Log

- ☐ Are potential changes to the event logs explicated and justified, and are the implications of these changes discussed? (Internal validity)
- ☐ Are the effects of missing data discussed? (Conclusion validity)

Model

- ☐ Is the construct "model" well-defined? (Completeness)
- ☐ Do all models adhere to their definition? (Soundness)

Model ⇆ Process

- ☐ Are potential representational biases and other limitations explicated? (Conclusion validity)

Experiment ⇆ Model

- ☐ Is comparability of the models ensured? (Conclusion validity)
- ☐ Are the effects of model transformation discussed? (Internal validity)
- ☐ Are the effects of missing model data discussed? (Conclusion validity)

Measure

- ☐ Does the design explicate the measures and describe how the values were obtained? (Direct and Conceptual replicability)
- ☐ Are the premises of the measures met? (Construct validity)

Measure ⇆ Construct

- ☐ Does each measure approximate the intended construct? (Construct validity)
- ☐ Are all necessary constructs covered by the measures? (Internal validity)

Experiment ⇆ Measure

- ☐ Are all values for all measures presented? (Internal validity)
- ☐ Are the effects of missing values discussed? (Conclusion validity)

3. Conclusion and Discussion

Hypothesis ⇆ Experiment

- ☐ Does acceptance or rejection of the hypotheses follow from the experiments' results? (Conclusion validity)
- ☐ Do the conclusions drawn from the experiment generalise beyond the experimental setting? (External validity)
- ☐ Are remaining non-mitigated threats to reliability and validity discussed?
- ☐ Are limitations of the experimental set-up discussed?

Claim

- ☐ Are implications of the claims discussed?

Fig. 5. The process discovery experimentation checklist.

However, using the checklist contributes to the transparency of experimentation, by promoting that they include enough detail to enable thorough review. As such, describing all relevant aspects in sufficient detail—as encouraged by the checklist—is a necessary though not sufficient condition for a reliable and valid result. To this end, in this section, we conduct an example experimentation using the checklist. That is, we follow the structure of the checklist and cover every item explicitly. First, we describe the context of the experiment by building the hypothesis in Section 5.1, after which we describe the design and execution of the experiment in Section 5.2. We finish with a conclusion and discussion of the experimental results in Section 5.3.

5.1 Hypotheses Building

☑ Claim. **Directly follows miner (DFM)** [47] is less sensitive to noise than SM [7].
☑ Concept. A *process* is defined as above [88]. An *event log* is a recording of executions of a process. *Noise* is recordings in an event log that do not (fully) correspond to behavior of the process the log was recorded from. *Sensitivity to noise* is the degree to which the result of an algorithm is influenced by noise in a recorded event log. We say an algorithm is *not sensitive to noise* if it can, independently of any noise, discover the process that is recorded in the event log. The *quality of an algorithm's result* is the degree to which this result fulfils a particular desirable property.
☑ Claim ⇆ Concept. Our claim only uses the concept of sensitivity to noise, which along with the auxiliary concepts (process, event log) required by it has been defined.
☑ Construct. We define an *event* as a record of an observation of the execution of an atomic process step, a *trace* as a totally ordered sequence of events, and an *event log* as a multiset of traces. For our purposes, a *model* is a workflow net [74]. We operationalize the quality of an algorithm's result in several *model quality properties*, each of which takes a model and perhaps an event log and returns a value indicating a particular quality aspect of that model. Traces are *noise* if they are contained in the event log, but cannot be replayed by the model that represents the process. As this model is unknown, we instead mimic noise as traces or events that are added to or removed from an event log after its recording. *Sensitivity to noise* is defined with respect to a model quality property: a process discovery technique is less sensitive to noise than another one if the quality is changing less for the technique than for the other technique under increasing noise levels.
☑ Concept ⇆ Construct. We assume that the traces that we generate during noise generation are not actually part of the process, as we do not know the processes involved in this experiment (threat to validity). We assume that the events as provided in the event log are totally ordered, as we lack knowledge of extraction procedures of publicly available data. The quality of a process discovery algorithm's result is often operationalized using standard quality constructs such as fitness, precision, generalization and simplicity. Our claim and hypothesis do not depend on these, thus we can abstract from these constructs here.
☑ Hypothesis. SM is less or equally sensitive to noise than DFM. This hypothesis could logically be disproved by a single counterexample, however will be addressed using statistical methods to obtain generality.
☑ Hypothesis ⇆ Claim. Our goal is to test the hypothesis. Therefore, we formulate one null hypothesis H0, which is derived from the claim: DFM is less sensitive to noise than SM. For testing, we use the counter hypothesis, which negates the claim. This means that if our hypothesis is falsified by statistical methods, we can conclude that the claim must hold. The hypothesis does not make any additional assumptions.
☑ Hypothesis ⇆ Construct. Our hypothesis uses only the construct of sensitivity to noise, which has been defined, along with the required auxiliary constructs.
☑ Claim ⇆ Algorithm. Our claim explicates the evaluated algorithms and does not generalize beyond these.

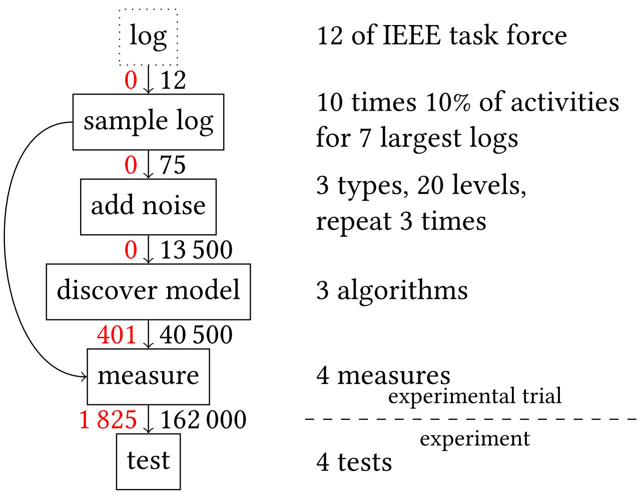

Fig. 6. Visual representation of our experimental trial setup. Annotated edges in black indicate how many combinations result, and in red indicate how many of these were errors.

5.2 Method Design and Execution

☑ Experiment.

▷ *Algorithm selection.* The hypothesis concerns two algorithms (SM and DFM), which are included in the experiment. A baseline is not necessary to validate the hypothesis but is included to position and interpret the results. That is, we include a **flower model (FM)**, which allows for any possible behavior of the activities in the event log, as long as these activities do not overlap in time. An alternative baseline would be a model iterating all traces of the event log, which would be prohibitively expensive for measures.

▷ *Design.* To test the sensitivity of algorithms to noise, we create a sample of noise levels (systematic) and quality criteria (convenience). Figure 6 shows an overview of the experiment setup. By generating noise using 20 levels of noise, we obtain 13,500 event logs. To each log, we apply the 3 discovery algorithms. Each model discovered model is assessed with respect to the original event log using 4 measures. We refer to the process of obtaining one measure as an experimental trial; there are 162,000 such trials.

To test our hypothesis, we apply the bootstrap method to these experimental trials: we take 10,000 re-samples with replacement and measure which algorithm is less sensitive on the re-sample. We reject our hypothesis if DFM is less sensitive in $\alpha * 10{,}000$ re-samples, with $\alpha = 0.05$. We correct for multiple tests being performed, that is, one test for each quality measure, using the Benjamini–Hochberg method [9]. The bootstrap method poses no assumptions on either the data or the computations; outliers need no special consideration. Applying the bootstrap method provides a quantification of the uncertainty present in the results and allows for a clear cut-off grounded in statistical theory. It is neither necessary to use test and training logs, nor to use k-fold cross-validation, as we do not compare the measures of the discovery algorithms with one another.

▷ *Missing data.* Not all experimental trials will be complete, due to empty event logs, missing models or missing measured values. As our noise-sensitivity measure is robust to missing trials, their number is rather low, and they will be shown to be well balanced over logs and models, we simply exclude those incomplete trials from the experiment.

▷ *Mitigated threats to reliability and validity.* We chose three different strategies for noise introduction, as these strategies might introduce bias. Each experimental trial was repeated three times, as noise introduction strategies are random and measures (precision) are conceptually not deterministic. Cryptographically secure randomness is not necessary, however a fixed seed was used for reproducibility reasons. Due to the reduced logs procedure (see below), the seven largest logs would contribute 10 times as much to the result as the other logs, so the experimental trials of the not-reduced logs were included 10 times in the experiments.

☑ Hypothesis ⇆ Experiment. The experiment design allows to test our hypothesis given a set of measures, a set of noise insertion strategies and set of event logs; randomization, repetition and the statistical test in the experiment otherwise increase generalizability. Even though the chosen measures are often used together in literature [12], the experiment does not provide for a way to generalize over the used measures; the hypothesis will be addressed for each measure individually. Nevertheless, we conjecture that the results hold validity beyond the used logs.

☑ Algorithm implementation. We assume the implementations for which links are provided in [47] (DFM) and [7] (SM). FM is a straightforward implementation.

☑ Algorithm ⇆ Algorithm implementation. We used publicly available implementations [7, 47], thus the implementations are correct if and only if they were correct in the papers where these algorithms were introduced.

☑ Algorithm implementation ⇆ Experiment. The experimental setup does not include computation time and no time-out was used for discovery techniques, thus a specification of computing hardware is not necessary.

The algorithms have parameters that may influence their handling of noise. Testing the algorithms' sensitivity to these parameters is not part of this experiment.

☑ Process. The organizations from which the logs were extracted cover several industries, including financial institutions, manufacturers, and insurance companies. The set of logs is a convenience sample from all possible event logs from all processes in all possible organizations. The nature of the processes plays no further role in this experiment.

☑ Event Log.

▷ *Selection.* We include all XES event logs published by the IEEE Task Force on Process Mining[4] as of 12 May 2021. These are BPIC2011, BPIC2013 closed problems, BPIC2013 open problems, BPIC2013 incidents, BPIC2015_1, BPIC2015_2, BPIC2015_3, BPIC2015_4, BPIC2015_5, BPIC2017, BPIC2018, BPIC2019. BPIC2016 was excluded as it involves UI data, which differs from process data in key characteristics and therefore is ill-suited for existing discovery algorithms.

▷ *Operationalization.* The obtained logs follow the XES-standard [1]. They were transformed into the conceptual definition used in this experiment using the `concept:name` attribute classifier. As a consequence, all activity instances are atomic and thus seen as to not overlap in time.

☑ Event Log ⇆ Process. The relation between log and process does not matter for this experimentation; knowledge of log extraction procedures is not necessary.

☑ Experiment ⇆ Event Log. The seven largest logs overwhelm model quality measures. Therefore, without further domain knowledge, these seven logs have been sampled by selecting a random group of 10% of the activities, and removing all events not related to these activities. Although this step limits the validity of the conclusions to smaller logs, the logs might have been filtered before publication.

[4]https://www.tf-pm.org/resources/logs

The noise introduction levels range from 0% to 95% in increments of 5%. The noise introduction strategies are: (1) insert event (of an activity randomly selected from a pool of 50 unused activities) in a trace; (2) delete event from a trace; (3) swap the order of two consecutive events. For each strategy, a separate log was constructed by applying the noise introduction to or after each event with the given likelihood independently. To mitigate random effects, each noise generation is repeated three times as part of the overall experiment design. If the activity sampling returned an empty log, the sampling was repeated. If the noise generation removed all events from a trace, the trace would still be valid and does not require adjustment.

☑ Model. SM returns models that are a subset of **Business Process Model and Notation (BPMN)** [54] models and can be translated to extended Petri nets [39], which establishes the formal semantics of this BPMN subset. DFM returns Directly Follows [47] models, which are formally defined in [47]. We assume all models returned by SM and DFM adhere to the definitions given in [7, 47].

☑ Model ⇆ Process. The logs are derived from real-life processes, which might not fit the representational biases of the algorithms. These mis-fits might make algorithms more susceptible to noise and potentially influence different algorithms differently. For this field-type experiment, we cannot mitigate this threat to validity.

☑ Experiment ⇆ Model. To enable comparisons, all discovered models are transformed into extended workflow nets [71] before being measured. For DFM, this transformation [47] is guaranteed, and the transformed models are sound (that is, one can execute every part and one can always reach the final state [74]). The subset of BPMN models returned by SM can be translated to extended Petri nets [39], by applying the ProM plug-in BPMN2PetriNetConverter and repair the final marking to ensure it consists of one token on the only place without outgoing transitions. There is no guarantee that these transformed nets are extended workflow nets nor that these models can reach their final marking. The model-quality measures require that each model is *easy sound*, i.e., the final marking *can* be reached [74]; all models in our experimentation were easy sound.

Missing data occurs if the discovery technique could fail to produce a model due to conceptual mismatches or program exceptions. For a failed discovery technique, we exclude the corresponding value for that measure (trial).

☑ Measure. There are two layers of measures: a sensitivity-to-noise measure and 4 model-quality measures.

▹ *Sensitivity to noise.* The hypothesis will be assessed using the sensitivity-to-noise measure, which is the absolute value of the slope of the linear model that best fits the measured quality values. This measure is deterministic.

▹ *Model quality.* To measure the quality of discovered models, we use four well-established measures: alignment-based fitness [4], ETC-precision [73], ETC-generalization [73], and “density” [50], adapted for Petri nets: $1 - a/(p * t * 2)$, where a, p, t are the numbers of arcs, places and transitions. It indicates the fraction of present over potential edges.

For our measures, we use alignments, whose computation requires the model to be able to terminate. As DFM guarantees soundness, all DFM models are conceptually compatible with alignments. Some SM models might be unsound; they are guaranteed to be deadlock-free but may have livelocks. If a model cannot terminate, the alignment computation simply times out, resulting in a missing experimental trial. The measures we use have no further restrictions and we treat them as black boxes.

The model quality measures are repeated three times as part of the overall experimental trial repetition. A time-out of 3 days was applied for each model quality measure; measures exceeding this were treated as missing experimental trials.

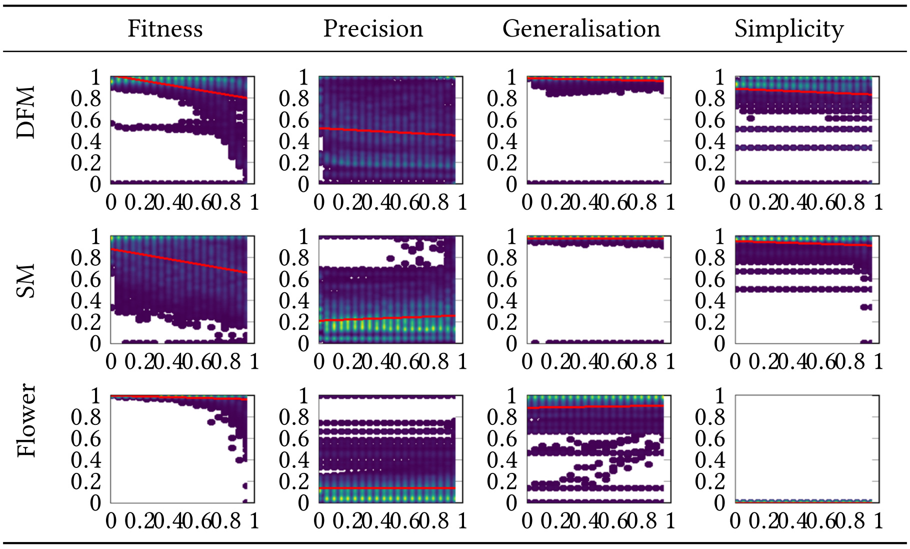

Fig. 7. Scatter density plots of our experiment on SM, DFM and Flower Miner; x-axis: noise level; y-axis: measure; red line: simple linear regression model; using the Viridis colour map (lighter colors indicate higher density).

☑ Measure ⇆ Construct. The sensitivity-to-noise measure is the slope of a fitted linear model to all experimental trials for a model quality measure and a discovery technique. The closer this slope is to 0, the less sensitive the technique is to noise. A linear relation might not be the most applicable relation between the level of noise and a model quality measure. However, it allows us to isolate the slope from the y-intercept easily, corresponding to sensitivity rather than absolute value. A visual inspection was performed to exclude obvious non-linear cases (Figure 7). The model quality measures have been shown to be associated with the quality of process models [12]. We chose a different simplicity measure than [12] as we require a normalized measure for the noise-sensitivity measure.

☑ Experiment ⇆ Measure.

▷ *Presentation.* The results are shown in Figure 7. The discarded measures have not been included. For illustrative purposes, red lines indicate the result of a simple linear regression overall data (without bootstrapping).

▷ *Missing values of measures.* For the measures, reasons for failure could be lack of models (due to errors in discovery), insufficient memory, unforeseen excessive run time, non-terminating models or program exceptions. For a failed measure, a tradeoff needs to be made: ideally, each log with at least one incomplete trial would need to be excluded from the results to avoid measuring the discovery algorithms on different data sets. However, removing such logs would introduce a bias towards the less complex logs (where complex may be different for each algorithm), thus reducing generalizability. In the experiment, 42 of the 75 sample logs had at least one missing measure. Of the 40,500 models, for 401 models not all measures could be computed, and a total of 1,825 measures is missing. The log with the highest number of missing measures had 194 measures missing (of 2,160, 4%). Therefore, we decided to keep the incomplete logs and accept that as a threat to validity, some values of measures are missing. All of the missing models were from SM—where DFM and flower discovered all of their models—and generally from the more complex sample logs.

5.3 Conclusion and Discussion

☑ Hypothesis ⇆ Experiment.

▹ *Acceptance of the hypothesis.* Based on the bootstrap tests, we reject the hypothesis for fitness: DFM is less sensitive to noise than SM. For the measures precision, generalization and simplicity, the experiment did not allow rejecting the hypothesis. However, from Figure 7, it is clear that for these measures, SM is less sensitive to noise than DFM.

The bootstrap tests were unanimous: the results were the same for all samples. Still, we argue that our approach is methodically necessary for conclusion validity: if one considered simple linear regression slopes directly (without repeated sampling in the bootstrap method), one could not establish whether two measured slopes are different "enough" to reach a conclusion. To put this into perspective: using the straightforward approach, the FM miner would have been the most stable for every measure except for generalization. This illustrates the inherent limitation of the generalization measure, as the FM is intuitively the most general model.

▹ *Generalization.* Generalization addresses external validity. In this experiment, we examined the claim with respect to different noise levels, meaning that we can generalize our findings to all noise levels in the examined logs. However, we did not systematically examine different log sizes or different levels of complexity. Therefore, we cannot claim any generalization for other logs. Obviously, the results do not generalize to other process discovery algorithms.

▹ *Remaining threats to reliability and validity.* We chose all currently publicly available data sets, as the choice of event logs might introduce bias; results may not generalize to all processes of all organizations. Missing values have been discussed before. Representational biases of the individual algorithms (see above) could be mitigated in a lab setting, but this would introduce other threats to validity. Noise introduction strategies might introduce changes that are not noise according to the (unknown) process behind a log, however the probability of this happening is presumably low.

▹ *Limitations.* The experimental setup did not include the seven largest logs, but reduced versions of these, as measures proved infeasible on these logs. Thus, the claim has effectively only been shown for smaller event logs. A sound hypothesis does not imply meaningful or watertight hypothesis: in this experiment, a constant insensible model would be correctly identified as insensitive to noise. That is, we did not test whether an algorithm was *better*, just whether it was more *stable*. The algorithms are instantiated with their default parameters; this might limit generalizability.

☑ Claim. This experiment might inform future process mining research and contribute to the ongoing discussion on evaluations in the field: rather than considering absolute values of model-quality measures, their sensitivity can be considered. Furthermore, it may inform end users in choosing discovery techniques, in case some knowledge on the noise level in an event log is available.

6 Discussion

In this section, we briefly discuss its contributions, limitations and (remaining) threats to validity of our research.

6.1 Contributions

This article is the result of long and careful deliberation, based on seminal literature and the existing principles of scientific enquiry. It contributes to process discovery research in theoretical (epistemological) and practical (methodical) respect. From a theoretical perspective, it establishes an understanding of process discovery as an empirical science in need of empirical research methods. By relating existing principles of scientific enquiry to existing process discovery research, we point out epistemological challenges in process discovery, such as the operationalization of constructs and the generalizability of results. Furthermore, the paper introduces a general terminological framework to be used as a shared vocabulary in developing research methods.

From a practical perspective, the paper provides concrete guidelines for researchers. Authors could consider the checklist as a guideline for conducting valid and reliable process discovery experiments. If checklist items are answered with more than yes or no, they may provide structure to ensure that each item has been sufficiently addressed. In addition, reviewers can use the checklist as a guideline for their assessment of the quality of the experiments in submitted papers. Using the checklist, it becomes easier to judge reliability and validity of the submission.

6.2 Limitations

Generalizability. In this article, we have shown the applicability of our research methodology in an exemplary experimentation, where we discuss how it guided the experimental design, execution and conclusion. However, this was only one example, meant to demonstrate and illustrate the applicability of the methodology. It does not show that the methodology is generalizable. Therefore, it needs to be applied for different experiments in different context, ideally by other researchers.

Evaluation. The demonstration of the methodology in an exemplary experiment also does not constitute a full evaluation. For that, we would need to show that the methodology fulfils its intended purpose: serving as a guideline for other researchers in conducting process discovery experiments. This aspect can only be evaluated by means of practical application in the field, conducted by researchers other than ourselves. Therefore, we invite the community to apply the methodology in their experiments and thereby to evaluate and advance it.

Completeness. The checklist does not claim to be complete. Given the practically endless options for conducting experiments, it would be impossible to list all potentially relevant questions. Instead, we consider the checklist to be a generic guideline, which allows researchers to reflect on how the quality and the contributions of research. When conducting an experiment, they need to adapt and potentially extend the checklist to fit their respective objectives.

6.3 Threats to Reliability and Validity

Usability. Our research methodology mainly consists of informal definitions and relations. The items of the checklist can only be addressed by careful scientific arguments in prose. As an example, consider the question "Do the claims use concepts correctly?". Authors may be tempted to answer this question with "yes" and move on. However, as we saw in our own experiment, answering this question requires a careful design of both the concept and the claim. In our exemplary evaluation, it took multiple iterations to grasp and operationalize the concept of "sensitivity to noise" in a way that it was specific enough to support our claim but general enough to apply to any model quality property. Especially for an inherently technical and traditionally formal research discipline like process discovery, this unusual way of conducting research might limit the practical usability of the methodology.

The usability of our methodology is also threatened by the page limitations of publication outlets, such as most conferences. As shown in Section 5, explicating all checklist items can become lengthy. However, all elements of the checklist are required to achieve sufficient reliability and validity of experimental evaluations. Therefore, it is not feasible to only consider a fraction of the checklist items. Instead, we propose authors to use the checklist to guide the design and execution of their experiment. This means that at least for themselves, they should have a good answer to each posed question and write it down in a separate document, which is provided, e.g., in a separate repository together with the implementation and the detailed results. For the actual publication, it then suffices to provide a brief synopsis, i.e., to write down the most important aspects of the experimental design and execution.

Guarantees. As we mention throughout the paper, our methodology is meant to support researchers in designing and conducting process discovery experiments. However, it cannot provide any guarantees about the quality of the research outcome. Even if researchers apply it to the best of their abilities, the resulting experiments might be unreliable or invalid. This is both a threat to reliability and to validity, just as the preceding one, originating from the practical aspirations of the experiments and can therefore not be fully mitigated.

Potential Bias. The design of the methodology is based on scientific argumentation. After drafting a first initial version of the terminology, the checklist, and the setting of the example experimentation, we described the exemplary evaluation through the lens of the checklist items, revising and refining all three parts in conjunction. We had long and intensive discussions that required us to reflect on our own research process, explicate our assumptions, and revise our understanding of what we thought was common knowledge. Therefore, this research approach can be subject to potential biases of the individual researchers. We mitigated these biases by including a set of researchers from different backgrounds and establishing a consensus amongst ourselves.

7 Conclusion

Process discovery algorithms aim to construct process models from event logs, such that these models can be analyzed to assist organizations in process optimization. In this article, we introduce a novel research methodology for process discovery: PDE. Having observed that a systematic approach for experimenting with process discovery algorithms is needed to develop process discovery into an empirical discipline, we applied the principles of scientific enquiry to the process discovery field. Based on our findings, we developed the methodology of PDE, which consists of a shared terminology and a checklist for experimentation. We demonstrated its applicability by means of an exemplary evaluation. The proposed methodology helps researchers to explicate the reliability and validity of experiments and to study the effects of these threats on the results of process discovery. We encourage the field to study the effect of other sources of potential threats to reliability and validity, such as ecological validity, i.e., the degree to which experimental results can be generalized to a real-world setting.

Many aspects of PDE may also be applicable to other process mining disciplines or research methods. For instance, elements of the shared terminology can also be found in conformance checking, which implies that the corresponding checklist items can be leveraged. Furthermore, other research methods, such as case studies, could benefit from our methodology. Examining and extending the applicability of PDE to other process mining disciplines and other research methods should therefore be a focus of future work.

The overall goal of our research is to improve reliability and validity of experimental results in process discovery. With the introduction of PDE, we propose a next step towards this goal. Given the complexities and idiosyncrasies of the process discovery field, it remains unclear whether external validity and conceptual replicability are realistically achievable for all experiments in the field. They might remain a scientific holy grail. However, this does not mean that we should not strive to achieve them. Only if our experiments are sufficiently reliable and valid, we can get closer to the ultimate goal of process discovery: to discover the true process.

Author Contributions

S.L. initiated the research; J.R., J.M.W., P.F., and S.L. conceived the research; J.R. and J.M.W. designed the research methodology; P.F. and S.L. evaluated the research methodology; S.L. performed the experiment; J.R. and J.M.W. wrote Sections 1–3 and 7; J.R.; S.L, and J.M.W. wrote Section 4; J.R. and S.L. wrote Sections 5 and 6; S.L. and P.F. reviewed the manuscript.

References

[1] Giovanni Acampora, Autilia Vitiello, Bruno Di Stefano, Wil van der Aalst, Christian Günther, and Eric Verbeek. 2017. IEEE 1849: The XES standard: The second IEEE standard sponsored by IEEE computational intelligence society [society briefs]. *IEEE Computational Intelligence Magazine* 12, 2 (2017), 4–8.

[2] Arya Adriansyah, Jorge Munoz-Gama, Josep Carmona, Boudewijn van Dongen, and Wil van der Aalst. 2013. Alignment based precision checking. In *Proceedings of the International Conference on Business Process Management Workshops*. Springer, 137–149.

[3] Arya Adriansyah, Jorge Munoz-Gama, Josep Carmona, Boudewijn van Dongen, and Wil van der Aalst. 2015. Measuring precision of modeled behavior. *Information systems and e-Business Management* 13, 1 (2015), 37–67.

[4] Arya Adriansyah, Boudewijn van Dongen, and Wil van der Aalst. 2011. Conformance checking using cost-based fitness analysis. In *Proceedings of the 2011 IEEE 15th International Enterprise Distributed Object Computing*. IEEE, 55–64.

[5] Adriano Augusto, Raffaele Conforti, Abel Armas-Cervantes, Marlon Dumas, and Marcello La Rosa. 2022. Measuring fitness and precision of automatically discovered process models: A principled and scalable approach. *IEEE Transactions on Knowledge and Data Engineering* 34, 4 (2022), 1870–1888.

[6] Adriano Augusto, Raffaele Conforti, Marlon Dumas, Marcello La Rosa, Fabrizio Maria Maggi, Andrea Marrella, Massimo Mecella, and Allar Soo. 2018. Automated discovery of process models from event logs: Review and benchmark. *IEEE Transactions on Knowledge and Data Engineering* 31, 4 (2018), 686–705.

[7] Adriano Augusto, Raffaele Conforti, Marlon Dumas, Marcello La Rosa, and Artem Polyvyanyy. 2019. Split miner: Automated discovery of accurate and simple business process models from event logs. *Knowledge and Information Systems* 59, 2 (2019), 251–284.

[8] Martin Bauer, Han van der Aa, and Matthias Weidlich. 2022. Sampling and approximation techniques for efficient process conformance checking. *Information Systems* 104 (2022), 101666.

[9] Yoav Benjamini and Yosef Hochberg. 1995. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B, Methodological* 57, 1 (1995), 289–300.

[10] Alessandro Berti, Sebastiaan van Zelst, and Wil van der Aalst. 2019. Process mining for python (PM4Py): Bridging the gap between process- and data science. arXiv:1905.06169. Retrieved from https://arxiv.org/abs/1905.06169

[11] Melike Bozkaya, Joost Gabriels, and Jan Martijn van der Werf. 2009. Process diagnostics: A method based on process mining. In *Proceedings of the International Conference on Information, Process, and Knowledge Management*, 22–27.

[12] Joos Buijs, Boudewijn van Dongen, and Wil van der Aalst. 2012. On the role of fitness, precision, generalization and simplicity in process discovery. In *Proceedings of the Conference on the Move to Meaningful Internet Systems*, 305–322.

[13] Joos Buijs, Boudewijn van Dongen, and Wil van der Aalst. 2014. Quality dimensions in process discovery: The importance of fitness, precision, generalization and simplicity. *International Journal of Cooperative Information Systems* 23, 1 (2014), 1440001.

[14] Josep Carmona, Boudewijn van Dongen, Andreas Solti, and Matthias Weidlich. 2018. *Conformance Checking*. Springer.

[15] Raffaele Conforti, Marcello La Rosa, and Arthur ter Hofstede. 2016. Filtering out infrequent behavior from business process event logs. *IEEE Transactions on Knowledge and Data Engineering* 29, 2 (2016), 300–314.

[16] Jonathan Cook and Alexander Wolf. 1998. Discovering models of software processes from event-based data. *ACM Transactions on Software Engineering and Methodology* 7, 3 (1998), 215–249.

[17] Jonathan Cook and Alexander Wolf. 1999. Software process validation: Quantitatively measuring the correspondence of a process to a model. *ACM Transactions on Software Engineering and Methodology* 8, 2 (1999), 147–176.

[18] Thomas Cormen, Charles Leiserson, Ronald Rivest, and Clifford Stein. 2009. *Introduction to Algorithms*. MIT Press.

[19] John Creswell. 2009. *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches* (3rd ed.). Sage Publications.

[20] Massimiliano de Leoni and Felix Mannhardt. 2015. Road Traffic Fine Management Process. Retrieved from https://data.4tu.nl/articles/dataset/Road_Traffic_Fine_Management_Process/12683249/1

[21] Jochen De Weerdt, Manu De Backer, Jan Vanthienen, and Bart Baesens. 2012. A multi-dimensional quality assessment of state-of-the-art process discovery algorithms using real-life event logs. *Information Systems* 37, 7 (2012), 654–676.

[22] Camil Demetrescu, Irene Finocchi, and Giuseppe Italiano. 2003. Algorithm engineering. *Bulletin of the EATCS* 79 (2003), 48–63.

[23] Fahame Emamjome, Robert Andrews, and Arthur ter Hofstede. 2019. A case study lens on process mining in practice. In *Proceedings of the Conference on the Move to Meaningful Internet Systems*, 127–145.

[24] Martin Enserink. 2018. Research on research. *Science* 361, 6408 (2018), 1178–1179.

[25] Mohammadreza Fani Sani, Sebastiaan van Zelst, and Wil van der Aalst. 2021. The impact of biased sampling of event logs on the performance of process discovery. *Computing* 103, 6 (2021), 1085–1104.

[26] Ulrich Frank. 2006. *Towards a Pluralistic Conception of Research Methods in Information Systems Research*. Technical Report 7. ICB-Research Report, Universität Duisburg-Essen.

[27] F. García, M. F. Bertoa, C. Calero, A. Vallecillo, F. Ruíz, M. Piattini, and M. Genero. 2006. Towards a consistent terminology for software measurement. *Information and Software Technology* 48, 8 (2006), 631–644.

[28] Kurt Gödel. 1930. *Über die Vollständigkeit des Logikkalküls*. Ph.D. Dissertation. University of Vienna.

[29] Stijn Goedertier, David Martens, Jan Vanthienen, and Bart Baesens. 2009. Robust process discovery with artificial negative events. *Journal of Machine Learning Research* 10 (2009), 1305–1340.

[30] E. Mark Gold. 1967. Language identification in the limit. *Information and Control* 10, 5 (1967), 447–474. DOI: https://doi.org/10.1016/S0019-9958(67)91165-5

[31] Stephan Guttinger. 2020. The limits of replicability. *European Journal for Philosophy of Science* 10, 2 (2020), 1–17.

[32] Alan Hevner. 2007. A three cycle view of design science research. *Scandinavian Journal of Information Systems* 19, 2 (2007), 4.

[33] Alan Hevner, Salvatore March, Jinsoo Park, and Sudha Ram. 2004. Design science in information systems research. *MIS Quarterly* 28, 1 (2004), 75–105.

[34] John Hooker. 1995. Testing heuristics: We have it all wrong. *Journal of Heuristics* (1995), 33–42.

[35] Gert Janssenswillen, Niels Donders, Toon Jouck, and Benoît Depaire. 2017. A comparative study of existing quality measures for process discovery. *Information Systems* 71 (2017), 1–15.

[36] Gert Janssenswillen, Toon Jouck, Mathijs Creemers, and Benoît Depaire. 2016. Measuring the quality of models with respect to the underlying system: An empirical study. In M. La Rosa, P. Loos, O. Pastor (Eds.), *Proceedings of the 14th International Conference on Business Process Management*, 73–89.

[37] Toon Jouck, Alfredo Bolt, Benoît Depaire, Massimiliano de Leoni, and Wil van der Aalst. 2018. An integrated framework for process discovery algorithm evaluation. arXiv:1806.07222.

[38] Martin Kabierski, Hoang Lam Nguyen, Lars Grunske, and Matthias Weidlich. 2021. Sampling what matters: Relevance-guided sampling of event logs. In *Proceedings of the International Conference on Process Mining*. IEEE, 64–71.

[39] Anna Kalenkova, Massimiliano de Leoni, and Wil van der Aalst. 2014. Discovering, analyzing and enhancing BPMN models using ProM. In L. Limonad and B. Weber (Eds.), *Proceedings of the BPM Demo Sessions*. Vol. 1295, CEUR-WS.org., 36–41.

[40] Bram Knols and Jan Martijn van der Werf. 2019. Measuring the behavioral quality of log sampling. In *Proceedings of the International Conference on Process Mining*, 97–104.

[41] Jelmer Koorn, Iris Beerepoot, Vinicius Stein Dani, Xixi Lu, Inge van de Weerd, Henrik Leopold, and Hajo Reijers. 2021. Bringing rigor to the qualitative evaluation of process mining findings: An analysis and a proposal. In *Proceedings of the International Conference on Process Mining*, 120–127.

[42] Thomas S. Kuhn. 1962. *The Structure of Scientific Revolutions* (50th ed.). University of Chicago Press.

[43] George Lakoff. 1985. *Women, Fire, and Dangerous Things*. University of Chicago Press, Chicago.

[44] Allen Lee and Richard Baskerville. 2003. Generalizing generalizability in information systems research. *Information Systems Research* 14, 3 (2003), 221–243.

[45] Sander Leemans, Dirk Fahland, and Wil van der Aalst. 2013. Discovering block-structured process models from event logs – A constructive approach. In *Proceedings of the International Conference on Application and Theory of Petri Nets and Concurrency (PETRI NETS '13).* J. M. Colom and J. Desel (Eds.), Lecture Notes in Computer Science, Vol. 7927, Springer, Berlin, 311–329.

[46] Sander Leemans, Dirk Fahland, and Wil van der Aalst. 2014. Discovering block-structured process models from incomplete event logs. In *Proceedings of the International Conference on Application and Theory of Petri Nets and Concurrency (PETRI NETS '14).* G. Ciardo and E. Kindler (Eds.), Lecture Notes in Computer Science, Vol. 8489. Springer, Cham. 91–110.

[47] Sander Leemans, Erik Poppe, and Moe Wynn. 2019. Directly follows-based process mining: exploration & a case study. In *Proceedings of the International Conference on Process Mining*. 25–32.

[48] Felix Mannhardt. 2016. Sepsis Cases - Event Log. Retrieved from https://data.4tu.nl/articles/dataset/Sepsis_Cases_Event_Log/12707639/1

[49] Laura Maruster, Ton Weijters, Wil van der Aalst, and Antal van den Bosch. 2006. A rule-based approach for process discovery: Dealing with noise and imbalance in process logs. *Data Mining and Knowledge Discovery* 13, 1 (2006), 67–87.

[50] Jan Mendling. 2007. *Detection and Prediction of Errors in EPC Business Process Models*. Ph.D. Dissertation. Wirtschaftsuniversität Wien.

[51] Jan Mendling, Benoît Depaire, and Henrik Leopold. 2021. Theory and practice of algorithm engineering. arXiv:2107.10675. Retrieved from https://arxiv.org/abs/2107.10675

[52] Jorge Munoz-Gama. 2016. *Conformance Checking and Diagnosis in Process Mining*. Springer.

[53] Jorge Munoz-Gama, Josep Carmona, and Wil van der Aalst. 2013. Conformance checking in the large: Partitioning and topology. In *Proceedings of the International Conference on Business Process Management*. F. Daniel, J. Wang, and B. Weber (Eds.), Lecture Notes in Computer Science, Vol. 8094. Springer, Berlin, 130–145.

[54] Object Management Group. 2011. Notation BPMN version 2.0.

[55] Ken Peffers, Tuure Tuunanen, Marcus Rothenberger, and Samir Chatterjee. 2007. A design science research methodology for information systems research. *Journal of Management Information Systems* 24, 3 (2007), 45–77.

[56] Roger Peng. 2011. Reproducible research in computational science. *Science* 334, 6060 (2011), 1226–1227.

[57] Artem Polyvyanyy, Andreas Solti, Matthias Weidlich, Claudio Di Ciccio, and Jan Mendling. 2020. Monotone precision and recall measures for comparing executions and specifications of dynamic systems. *ACM Transactions on Software Engineering and Methodology* 29, 3 (2020), 1–41.

[58] Karl Popper. 1935. *Logik der Forschung – Zur Erkenntnistheorie der modernen Naturwissenschaften.*

[59] Jan Recker. 2021. *Scientific Research in Information Systems* (2nd ed.). Springer.

[60] Jana-Rebecca Rehse and Peter Fettke. 2018. Process mining crimes – A threat to the validity of process discovery evaluations. In *Proceedings of the International Conference on Business Process Management Forum*, 3–19.

[61] Jana-Rebecca Rehse, Peter Fettke, and Peter Loos. 2018. Process mining and the black swan: An empirical analysis of the influence of unobserved behavior on the quality of mined process models. In *Proceedings of the International Conference on Business Process Management Workshops*, 256–268.

[62] Joel Ribeiro, Josep Carmona, Mustafa Misir, and Michele Sebag. 2014. A recommender system for process discovery. In *Proceedings of the International Conference on Business Process Management*, 67–83.

[63] Anne Rozinat, Ana Karla Alves de Medeiros, Christian Günther, Ton Weijters, and Wil van der Aalst. 2007. *Towards an Evaluation Framework for Process Mining Algorithms.* BPM Center Report BPM-07-06.

[64] Anne Rozinat, Ana Karla Alves de Medeiros, Christian Günther, Ton Weijters, and Wil van der Aalst. 2008. The need for a process mining evaluation framework in research and practice. In *Proceedings of the International Conference on Business Process Management Workshops*, 84–89.

[65] Per Runeson and Martin Höst. 2009. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering* 14, 2 (2009), 131–164.

[66] Peter Sanders. 2009. Algorithm engineering – An attempt at a definition. In *Proceedings of the International Conference on Efficient Algorithms.* S. Albers, H. Alt, and S. Näher (Eds.), Lecture Notes in Computer Science, Vol. 5760, Springer, Berlin, 321–340.

[67] William Shadish, Thomas Cook, and Donald Campbell. 2002. *Experimental and Quasi-Experimental Designs for Generalized Causal Inference.* Wadsworth Cengage Learning.

[68] Peter Swanborn. 1996. A common base for quality control criteria in quantitative and qualitative research. *Quality & Quantity* 30, 1 (1996), 19–35.

[69] Anja Syring, Niek Tax, and Wil van der Aalst. 2019. Evaluating conformance measures in process mining using conformance propositions. In *Proceedings of the Transactions on Petri Nets and Other Models of Concurrency XIV*. Maciej Koutny, Lucia Pomello, and Lars Michael Kristensen (Eds.), Lecture Notes in Computer Science, Vol. 11790, Springer, 192–221.

[70] Niek Tax, Xixi Lu, Natalia Sidorova, Dirk Fahland, and Wil van der Aalst. 2018. The imprecisions of precision measures in process mining. *Information Processing Letters* 135 (2018), 1–8.

[71] Wil van der Aalst. 1997. Verification of workflow nets. In *Proceedings of the International Conference on Application and Theory of Petri Nets.* P. Azéma and G. Balbo (Eds.), Lecture Notes in Computer Science, Vol. 1248. Springer, Berlin, 407–426.

[72] Wil van der Aalst. 2016. *Process Mining - Data Science in Action* (2nd ed.). Springer.

[73] Wil van der Aalst, Arya Adriansyah, and Boudewijn van Dongen. 2012. Replaying history on process models for conformance checking and performance analysis. *Data Mining and Knowledge Discovery* 2, 2 (2012), 182–192.

[74] Wil van der Aalst, Kees van Hee, Arthur ter Hofstede, Natalia Sidorova, Eric Verbeek, Marc Voorhoeve, and Moe Wynn. 2011. Soundness of workflow nets: classification, decidability, and analysis. *Formal Aspects of Computing* 23, 3 (2011), 333–363.

[75] Wil van der Aalst, Ton Weijters, and Laura Maruster. 2004. Workflow mining: Discovering process models from event logs. *Knowledge & Data Engineering* 16, 9 (2004), 1128–1142.

[76] Jan Martijn van der Werf, Artem Polyvyanyy, Bart van Wensveen, Matthieu Brinkhuis, and Hajo Reijers. 2021. All that glitters is not gold: Towards process discovery techniques with guarantees. In *Proceedings of the 33rd International Conference on Advanced Information Systems Engineering*, 141–157.

[77] Jan Martijn van der Werf, Boudewijn van Dongen, Cor Hurkens, and Alexander Serebrenik. 2008. Process discovery using integer linear programming. In *Proceedings of the International Conference on Applications and Theory of Petri Nets*, 368–387.

[78] Boudewijn van Dongen. 2024. BPI Challenge. Retrieved from https://data.4tu.nl/

[79] Boudewijn van Dongen, Ana Karla Alves de Medeiros, Eric Verbeek, Ton Weijters, and Wil van der Aalst. 2005. The ProM framework: A new era in process mining tool support. In *Proceedings of the International Conference on Applications and Theory of Petri Nets*, 444–454.

[80] Maikel van Eck, Xixi Lu, Sander Leemans, and Wil van der Aalst. 2015. PM^2: A process mining project methodology. In *Proceedings of the International Conference on Advanced Information Systems Engineering*, 297–313.

[81] Sebastiaan van Zelst, Boudewijn van Dongen, Wil van der Aalst, and Eric Verbeek. 2018. Discovering workflow nets using integer linear programming. *Computing* 100, 5 (2018), 529–556.

[82] Seppe vanden Broucke and Jochen De Weerdt. 2017. Fodina: A robust and flexible heuristic process discovery technique. *Decision Support Systems* 100 (2017), 109–118.

[83] Seppe vanden Broucke, Jochen De Weerdt, Jan Vanthienen, and Bart Baesens. 2013. A comprehensive benchmarking framework (CoBeFra) for conformance analysis between procedural process models and event logs in ProM. In *Proceedings of the Symposium on Computational Intelligence and Data Mining*. IEEE, 254–261.

[84] Seppe vanden Broucke, Jochen De Weerdt, Jan Vanthienen, and Bart Baesens. 2014. Determining process model precision and generalization with weighted artificial negative events. *IEEE Transactions on Knowledge and Data Engineering* 26, 8 (2014), 1877–1889.

[85] Philip Weber, Behzad Bordbar, Peter Tiňo, and Basim Majeed. 2011. A framework for comparing process mining algorithms. In *Proceedings of the GCC Conference and Exhibition.* IEEE, 625–628.

[86] Ton Weijters and Joel Ribeiro. 2011. Flexible heuristics miner (FHM). In *Proceedings of the Symposium on Computational Intelligence and Data Mining*, 310–317.

[87] Ton Weijters, Wil van der Aalst, and Ana Karla Alves de Medeiros. 2006. *Process Mining with the Heuristics Miner-Algorithm.* Technical Report 166. Technische Universiteit Eindhoven.

[88] Mathias Weske. 2019. *Business Process Management: Concepts, Languages, Architectures* (3rd ed.). Springer.

[89] Roel Wieringa. 2014. *Design Science Methodology for Information Systems and Software Engineering*. Springer.

[90] Christopher K. I. Williams. 2021. The effect of class imbalance on precision-recall curves. *Neural Computation* 33, 4 (04 2021), 853–857. DOI: https://doi.org/10.1162/neco_a_01362

Received 22 November 2022; revised 5 December 2023; accepted 12 May 2024