A Stability Assessment Framework for Process Discovery Techniques

Pieter De Koninck(✉) and Jochen De Weerdt

Research Centre for Management Informatics Faculty of Economics and Business, KU Leuven, Leuven, Belgium
{pieter.dekoninck,jochen.deweerdt}@kuleuven.be

Abstract. An extensive amount of work has addressed the evaluation of process discovery techniques and the process models they discover based on concepts like fitness, precision, generalization and simplicity. In this paper, we claim that stability could be considered as an important supplementary evaluation dimension for process discovery next to accuracy and comprehensibility, with ties to the generalization concept. As such, our core contribution is a new framework to measure stability of process discovery techniques. In this paper, the design choices of the different components of the framework are explained. Furthermore, using an experimental evaluation involving both artificial and real-life event logs, the appropriateness and relevance of the stability assessment framework is demonstrated.

Keywords: Stability · Process discovery · Conformance checking · Validity · Log perturbation

1 Introduction

In unsupervised learning, where there is no straightforward way to evaluate discovered solutions, an important question is whether or not a specific solution is valid [12]. Common domains of unsupervised learning are clustering, latent variable methods such as Gaussian Mixture Models, and certain neural network models such as Self-Organizing Maps. Applications can be found in bioinformatics, data mining and pattern recognition, among others. Process discovery, i.e. the automated construction of process models from event logs, is essentially an unsupervised learning task as well. Admittedly, discovered process models can be evaluated structurally, e.g. on soundness [21], or based on the event log through conformance checking (for an overview see [7]). Nonetheless, there is no strict variable or label to predict, hence the discovery of a process model should be considered an unsupervised learning task.

The importance of validity of unsupervised learning algorithms is addressed in [12] as follows: '*It is difficult to ascertain the validity of inferences drawn from the output of most unsupervised learning algorithms. One must resort to heuristic arguments not only for motivating the algorithms, as is often the case in* *supervised learning as well, but also for judgements as to the quality of the results. This uncomfortable situation has led to heavy proliferation of proposed methods, since effectiveness is a matter of opinion and cannot be verified directly.*'.

While validity of process discovery techniques is partially addressed by the whole plethora of conformance checking techniques, it is argued in this paper that evaluation of process discovery algorithms lacks a thorough methodology to assess the stability of these algorithms. In clustering research, a stability-based assessment of validity has received plenty of attention [12,17]. It is argued in this paper that this stability dimension is missing to a large extent and should complement the already understood evaluation dimensions in process discovery, i.e. accuracy (recall, precision, generalization) and comprehensibility (simplicity).

Conceptually, the stability of a process discovery technique can be defined as *the consistency of process discovery solutions obtained using this technique from perturbed sets of input data or settings*. This consistency is measured as the similarity between the discovered process models for each of the perturbed data sets or deviating settings. The different constructs will be elaborated on in Sect. 2. Stability as a dimension could be further refined into different types, for instance log perturbation stability or parameter stability, depending on the method of perturbation (perturbing the event log versus perturbing the parameter settings of the discovery technique). Most likely, there is an interdependency between both: the same technique with different parameter settings could be more stable with regards to log perturbations. Parameter sensitivity of process discovery techniques has been partially addressed in [2], based on fitness and precision rather than stability.

Observe that log-perturbation stability, as it is defined and constructed here, is conceptually related to existing dimensions for the evaluation of discovered process models, specifically generalization. Generalization is defined as measuring the probability that, given an event log and a process model, a next batch of process instances not in the original event log will invalidate a process model [20]. For an overview and evaluation of existing generalization metrics, we refer to [23]. The log-perturbation could be seen as a variation on this 'next batch' of behaviour, and similarity between the discovered process model could be seen as a quantification of the validity of baseline process model.

In this paper, we propose a new framework for measuring the stability of process discovery techniques through a stability index, inspired by the approach in [15]. Although the framework could be adapted for measuring different types of stability, we focus on log perturbation stability, which is a variant with regards to repeatedly resampled or perturbed input data. The framework has been implemented as a ProM-plugin[1] and can be used by process miners to assess the stability of different process discovery techniques, given an event log of their interest.

Given this objective, the rest of this paper is structured as follows: in Sect. 2, a general approach for assessing stability is proposed. In Sect. 3, the approach

[1] The plugin, screenshots and additional information can be found at http://www.processmining.be/PDStability/.

is extensively evaluated considering two applications: evaluating the stability of discovered process models with regards to correctness, and evaluating the stability of process models with regards to completeness. Finally, in Sect. 4, conclusions are formulated and an outlook to future adaptations is presented.

2 A Stability Assessment Framework

The approach proposed in this paper is based on a methodology for stability-based validation of clustering solutions in [14], which was adapted for biclustering solutions in [15]. As discussed in Sect. 1, clustering is one of the most well-known unsupervised learning techniques. As such, it suffers from the same issues regarding solution validity as other unsupervised learning techniques.

In [14,15], resampling/perturbation strategies, learning algorithms, and solution similarity metrics are proposed that are specifically designed for (bi)clustering problems. In this domain, stability was shown to be an effective metric for assessing the validity of a clustering solution, e.g. with regards to cluster size or clustering technique. An advantage of the clustering domain as compared to process discovery is the existence of alternative validity indices, such as entropy or gap statistics, which can be used as a reference for comparing a stability-based approach. In process discovery, solution validity is a more complex construct, since it is already partially addressed by existing metrics. Specifically, our approach is related to the generalization sub-dimension of accuracy, as explained in the previous section. Nonetheless, we claim that a supplementary dimension to existing interpretations of process discovery validity should be considered, i.e. stability.

As such, this paper contributes by proposing a stability assessment framework for process discovery techniques. The framework is tailored to measure so-called 'log perturbation stability', however it can be reconfigured for assessing other types of stability, such as parameter stability.

In Fig. 1, our stability assessment framework is depicted. Tailoring the framework to process discovery entails the configuration of three main components, i.e. the perturbation strategy (step 1), the solution similarity computation (step 3), and a stability index calculation (step 4). In addition, a process discovery technique should be chosen (step 2).

The steps of our approach thus become:

1. **Step 1**: Given an event log L, and a log perturbation function $P()$, create n perturbed versions of the event log: $P_1(L)$ to $P_n(L)$.
2. **Step 2**: Discover a process model PM by applying a process discovery technique $PD()$ to the original event log: $PM = PD(L)$ and to the perturbed event logs: $PM_i = PD(P_i(L))$ with $i \in \{1..n\}$.
3. **Step 3**: Given a similarity index $I(PM_x, PM_y)$, quantify the similarity between the discovered process model on the original dataset and the discovered process model on the perturbed dataset as $I(PM, PM_i)$.

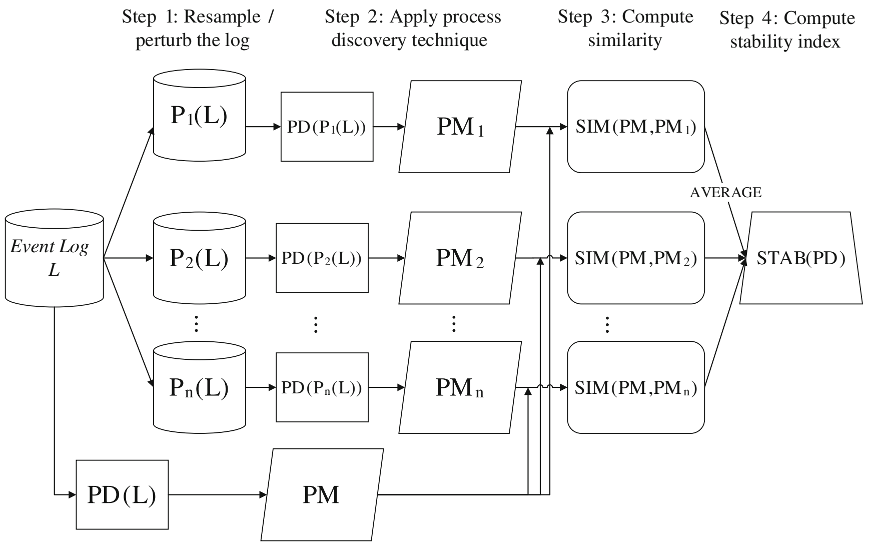
Step 1: Resample / perturb the log
Step 2: Apply process discovery technique
Step 3: Compute similarity
Step 4: Compute stability index
Event Log L
P1(L)
P2(L)
Pn(L)
PD(P1(L))
PD(P2(L))
PD(Pn(L))
PM1
PM2
PMn
SIM(PM,PM1)
SIM(PM,PM2)
SIM(PM,PMn)
AVERAGE
STAB(PD)
PD(L)
PM

Fig. 1. A visualization of the proposed approach for calculating the stability of a discovered process model, based on a similar diagram in [15].

4. **Step 4**: Average these similarity measures to create a stability metric for event log L and discovery technique $PD()$ as

$$S^{PD} = \frac{1}{n}\sum_{i=1}^{n} I(PM, PM_i) \tag{1}$$

Observe that a higher value for S^{PD} indicates a better stability of the solution. As such, this metric can be used for evaluating a process discovery outcome. In the remainder of this section, we describe the three main components of our framework: a perturbation strategy based on resampling and noise induction (Sect. 2.1), computation of solution similarity based on process model similarity metrics (Sect. 2.2), and calculation of the stability index based on a window-based approach (Sect. 2.3).

2.1 Step 1: Log Perturbation Strategy

Perturbing event logs essentially boils down to three options: either some behaviour is removed, or some behaviour is added, or a combination of both. There are many different ways to do this. Regarding the removal of behaviour, event log perturbation can be approached through case-level resampling in a random fashion, which is closely related to classical bootstrapping [5]. Note that case-level bootstrapping an event log becomes trace-level bootstrapping. When dealing with event logs, an important consideration is whether to bootstrap process instances or distinct process instances (i.e. the effect of imbalance on the bootstrap sample). An alternative to random resampling is systematic leave-one-out cross-validation. Cross-validation has been proposed briefly in [20], in the context of generalization for process-mining techniques, where generalization is measured by leave-one-out cross-validation as follows: leave out one process instance, and count the percentage of instances that can still be replayed on the discovered process model. Observe that our approach deliberately does not incorporate any form of replay.

Secondly, regarding the addition of behaviour, small perturbations of event logs strongly relates to the idea of adding noise to the log. In [18], four types of noise were initially defined: remove head, remove tail, remove body, and swap tasks. In [6], the removal of a single task was added as a noise induction scheme, together with the combination of all previous noise types. These noise induction types were already used to evaluate robustness of process discovery techniques, for instance in [11]. However, in contrast to our paper, this work evaluated the robustness to noise of process discovery techniques directly based on traditional accuracy metrics. Here, we propose a framework for assessing the stability of process discovery techniques that is independent from the actual accuracy or comprehensibility of the outcome. As such, it is an orthogonal evaluation dimension that should be taken into account.

Taking these aspects into consideration, the log perturbation strategy underlying our stability assessment framework is as follows. First behaviour can be removed through a resampling procedure, which is essentially undersampling at the level of distinct process instances. However, to make the resampling a bit less naive, the probability that a distinct process instance is removed, is inversely proportional to the frequency with which this distinct process instance is present in the event log. Secondly, behaviour can be added through noise induction. Albeit the several noise types already available [18], we opt to include three types of noise: remove a single event, swap two events, and add a random single event (from the log activity alphabet) at a random place in the process instance. Noise addition is performed at process instance level. For both removal (undersampling at distinct process instance level) and addition (noise induction at process instance level), a percentage of affected instances should be chosen. Observe that in case both perturbation options are applied, the resampling is performed first and the noise induction is applied second.

2.2 Step 3: Solution Similarity Computation

An extensive overview of similarity metrics for the pairwise comparison of business process models is presented in [8]. Three distinct categories of similarity metrics are proposed: first, node matching similarity, where similarity is based on the labels and attributes attributed to the different elements of a process model; secondly, structural similarity, where the labels of these elements are compared as well as the topology of the process models; and thirdly, behavioural similarity, where the labels of the elements are compared as well as causal relations captured in the process model.

Given our context of process discovery from an event log, node matching similarity is irrelevant. However, both structural as well as behavioural similarity metrics can be of use within the stability assessment framework. Regarding structural similarity, a so-called graph edit distance similarity is defined in [8], based on the amount of insertions and deletions that are necessary to transform one process graph into the other. Other structural metrics such as tree edit distance are available as well [1]. Looking at behavioural similarity, a common approach relies on causal footprints [9,10], referred to as the causal footprint similarity. Other options are transition adjacency-based similarity [29], or behavioral profile-based similarity [13,24,25]. A final category of process model similarity can be described as event-log based. Such metrics are based on the principle that not all pathways in a process model are equally important, and that behaviour that is more likely given the event log should be represented as such in a similarity metrics. Examples can be found in [19].

The current stability assessment framework incorporates three similarity metrics: (1) Graph-edit Distance (GED) [8], (2) causal footprint-based similarity (CF) [9], and (3) behavioural profile-based similarity (BP) [24]. In Sect. 3, the suitability of these metrics is assessed in an experimental evaluation.

2.3 Step 4: Stability Index Computation

Finally, in step 4 of our framework, the stability index is computed as an average over a number of iterations, as detailed in Algorithm 1. Hereto, three extra input parameters are necessary: a minimal number of iterations r_{min}, a review window Δr and a maximal stability error ϵ_S. Typical values for these parameters are 20, 10, and 0.005 respectively. This iterative approach serves a double purpose: on the one hand, it ensures that the final stability is robust and sufficiently precise, by enforcing an upper bound on the stability error; on the other hand, it prevents unnecessary computation, by terminating once the stability error is sufficiently small and the minimal number of iterations have been performed.

3 Experimental Evaluation

In this section, the configurations of the proposed stability assessment framework are analyzed, together with an investigation of the effects of the level of perturbation added and the specific characteristics of an event log. The objective is to show the appropriateness of the proposed constructs for the evaluation of process discovery techniques. Therefore, this section is structured as follows: first, in Sect. 3.1, the global setup of the evaluation is discussed, with regard to the datasets, process discovery techniques, similarity metrics and perturbation strategies used. Section 3.2 discusses the effect of the level of perturbation to which the event log is exposed. Section 3.3 provides the global results, while Sect. 3.4 takes a closer look at the effects of the characteristics of the event log on the stability.

Algorithm 1. Stability evaluation

```
Input: $L$ := Event log, $PD$ := Process discovery algorithm, $P$:= Perturbation strategy, $s$ := similarity metric;
Input: $r_{min}$ := 20, $\Delta r$ := 10, $\epsilon_S$ := 0.005; % Configuration
Output: $S$ := Stability measure for the combination of event log $L$ and discovery algorithm $PD$

1: function STABILITY( $L$, $PD$, $P$, $s$,$r_{min}$, $\Delta r$, $\epsilon_S$ )
2:     $r := 1$ % Iteration
3:     $PM := PD(L)$ % Baseline discovered process model
4:     $u() := \{\}$ % List of similarity results per iteration
5:     $w() := \{\}$ % List of stability results per iteration
6:     while $(r < r_{min}) \vee [max_{p,q}|w(p) - w(q)| > \epsilon_S; \forall p, q : r - \Delta r < p < q \leq r)]$ do
7:         $L_r := P_r(L)$ % Perturb the log
8:         $PM_r := PD(L_r)$ % Discovered process model from perturbed log
9:         $u(r) := s(PM, PM_r)$ % Calculate similarity with baseline model
10:        $w(r) := \frac{(r-1)*w(r-1)+u(r)}{r}$ % Calculate stability
11:        $r := r + 1$
12:    end while
13:    return $S := w(r - 1)$
14: end function
```

3.1 Setup

Five aspects of the experimental setup are of interest: the effect of characteristics of the event log on stability, the differences regarding process discovery techniques with regards to stability, the similarity metric used to compute the stability, the chosen perturbation strategy, and the level of perturbation induced by this strategy.

Firstly, the event log characteristics. We have set up experiments with 20 artificial event logs, as shown in Table 1. These datasets are taken from [3], to make our results compatible with other findings in the process mining domain. With regards to the characteristics of the event log, three measures are under scrutiny: the number of distinct process instances, the number of distinct events in the log, and the average number of events per process instance. The different activity structures on the underlying process models leveraged to create the artificial logs in [3], such as the presence loops of length 1 or 2, arbitrary or structured loops, invisible tasks are not considered here, since we are not concerned with rediscovering the artificial process model. As shown in Table 1, these characteristics vary sufficiently across the different event logs. Furthermore, we have repeated our setup on 5 real-life event logs [7], also listed in Table 1, to test whether similar results can be found using realistic event logs.

Secondly, the 8 process discovery techniques that are included in our study are the following, with default settings, and converted to Petri Nets where necessary: (1) Alpha miner [21], (2) Alpha++ miner [27], (3) Fodina [22], (4) Heuristics Miner [26], (5) ILPMiner [28], (6) Inductive Miner [16], (7) Flower miner, a technique that produces an underfitting flower model; (8) Naive [22], a discovery technique that naively models a connection between two transitions if they ever follow each other directly in the event log, unless these events overlap in time, in which case a connection is made to the closest non-overlappping transition.

Thirdly, three similarity metrics were used for comparing models discovered from perturbed event logs to the baseline discovered model, as described

Table 1. Characteristics of the artifical and real-life event logs used for the evaluation: number of process instances (**#PI**), distinct process instances (**#DPI**), number of different events (**#EV**) and average number of events per process instance ($\frac{\#EV}{PI}$).

Logname	#PI	#DPI	#EV	#EV/PI	Artificial	Real-life
grpd_g22pi300	300	24	26	10.32	✓	
groupedFollowsl1l_500	500	29	8	11.79	✓	
grpd_g19pi300	300	32	25	13.69	✓	
grpd_g13pi300	300	35	24	16.69	✓	
grpd_g12pi300	300	38	28	16.14	✓	
grpd_g24pi300	300	46	23	13.77	✓	
grpd_g4pi300	300	48	31	19.92	✓	
grouped_g2pi300	300	65	24	15.00	✓	
grpd_g5pi300	300	66	22	20.57	✓	
driveClass_700	700	87	13	21.00	✓	
grpd_g6pi300	300	92	25	18.06	✓	
grpd_g9pi300	300	102	28	18.93	✓	
groupedFollowsparallel5_700	700	109	12	12.00	✓	
grpd_g10pi300	300	110	25	13.72	✓	
grpd_g15pi300	300	135	27	13.26	✓	
herbstFig6p37_700	700	135	20	20.00	✓	
grpd_g14pi300	300	157	26	37.80	✓	
grpd_g20pi300	300	187	23	20.64	✓	
grpd_g7pi300	300	231	31	48.17	✓	
grpd_g3pi300	300	239	31	48.66	✓	
MOA	2004	71	49	6.20		✓
KP2P	10487	76	23	9.33		✓
ICP	6407	155	18	5.99		✓
MCRM	956	212	22	11.73		✓
KIM	1541	251	18	5.62		✓

in Sect. 2.2: Graph-edit Distance [8], causal footprint based similarity [9], and behavioural profile based similarity [24]. The latter is measured as a weighted sum between exclusiveness similarity, order similarity, interleaving order similarity, extended order similarity and extended interleaving similarity, as it was implemented in *JBPT*-library.

Fourthly, three different strategies for generating perturbations are considered here. On the one hand, a resampling method, where $p\,\%$ distinct process instances are randomly removed from the event log. The probability of removal is the inverse of the frequency of that distinct process instance in the event log. On the other hand, a noise induction method, where $q\,\%$ process instances have one random event removed, added or two events swapped. One of these three perturbations is randomly chosen with equal probabilities. Finally, a third setting is included that combines both methods: first, p % distinct process instances are removed; second, q % process instances have one random event removed, added or two events swapped. Both strategies are tested with p and/or q equal to 10 %.

Finally, a small note on the specific calculations and environment of the experiments: all experiments were run on a Intel XEon E5-2699 v3 processor of a Windows Server 2012 R2. For the calculation of stability, 20 fixed iterations where taken, rather than the adaptive strategy described in Algorithm 1. The duration of 1 set of specifications (i.e. 20 iterations) was restricted to 10 min. Of the 480 configurations on artificial logs that were evaluated using three different similarity metrics, 62 did not finish the mining task within the time limit: 44 combinations with Alpha++, 16 combinations of ILP, 1 combination with Fodina and 1 combination with Inductive miner. On the real-life datasets, 120 configurations were tested of which 9 resulted in a timeout: three configurations with Alpha++ and two with Alpha, Fodina and ILP.

3.2 Effect of the Percentage of Perturbation

Figures 2 and 3 show the effects of varying percentages of noise, when using Causal Footprint similarity as an underlying metric and, respectively, Heuristics Miner and Alpha miner as process discovery algorithm. The points represent average stability over 5 of the artificial event logs. A couple of observations can be made from these figures. First, observe that, as expected, the average stability declines as it is exposed to higher percentages of noise, all other things equal. The same observation holds for resampling percentages, at least when combined with a noise percentage smaller than 40 %. Secondly, remark that the stability appears to be a lot more sensitive to the level of noise than the level of resampling. This observation should be kept in mind regarding the results in Sect. 3.3, as the same percentage of noise induction has a greater effect than the resampling. Thirdly, observe that the curve for the results using Heuristics miner (Fig. 2) declines less rapidly than the one using Alpha miner (Fig. 3). Finally, even at high levels of noise induction, the resulting stability when using Heuristics miner lies around 0.85, whereas the resulting stability using Alpha miner performs significantly worse, even at lower percentages of noise.

3.3 Results of the Experimental Evaluation

Before interpreting the results, a note should be made regarding the compatibility of the similarity metrics and some of the process discovery techniques. The technique based on behavioural profiles, for example, requires that there are no unconnected transitions. Some techniques, however, do not guarantee that no unconnected transitions will be mined. Therefore, in some combinations with BP, no result could be obtained. Specifically, for the artificial logs, this is the case in 33 configurations with Alpha miner, 12 configurations with ILP, and 2 configurations with Alpha++. For the real-life logs, this was the case in 11 combinations of Alpha with BP.

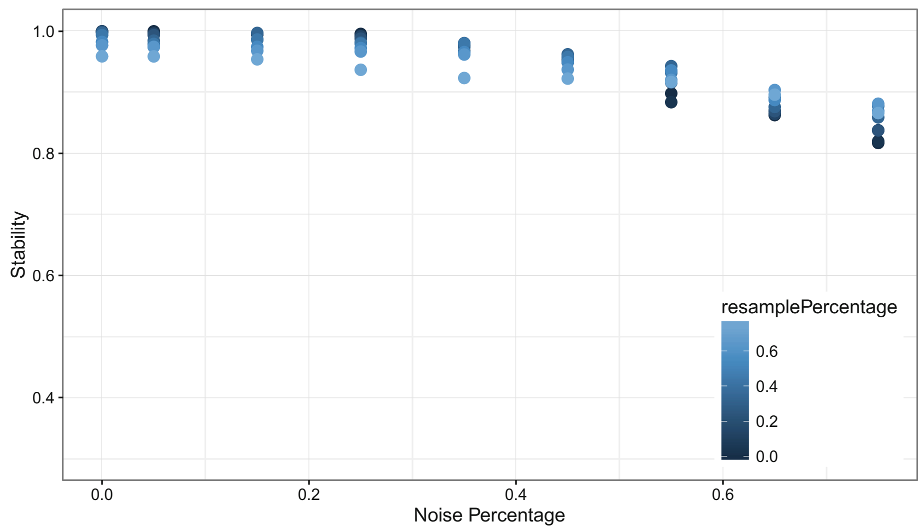

Fig. 2. Scatterplot based on noise percentage of the average stability over 5 artificial event logs using Heuristics Miner and a stability based on Causal Footprints, for resampling and noise induction percentages equal to 0 % or ranging from 5 to 75 % with 10 % intervals

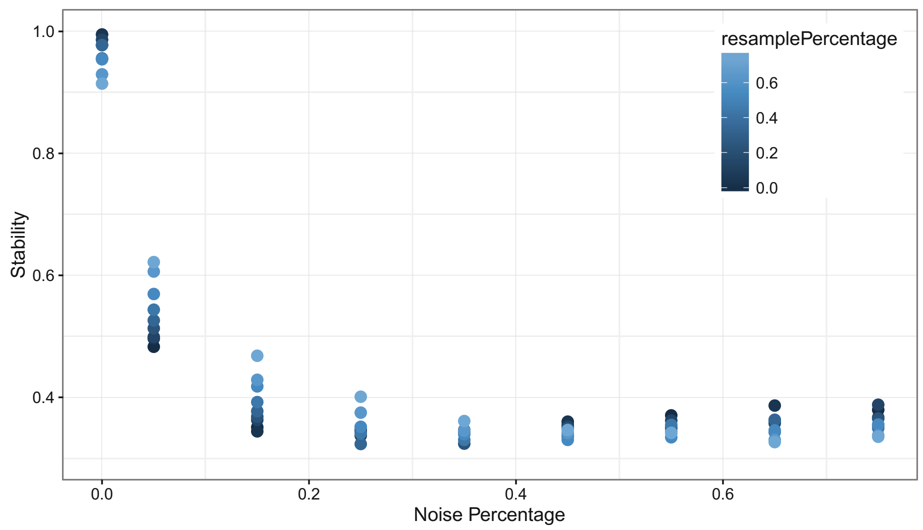

Fig. 3. Scatterplot based on noise percentage of the average stability over 5 artificial event logs using Alpha Miner and a stability based on Causal Footprints, for resampling and noise induction percentages equal to 0 % or ranging from 5 to 75 % with 10 % intervals

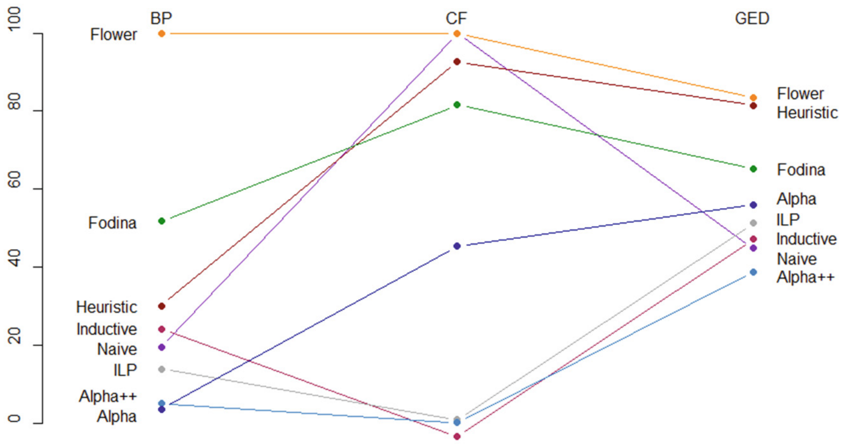

Fig. 4. Visualization of the average stability results when applying a noise induction strategy. Averages over 20 artificial event logs.

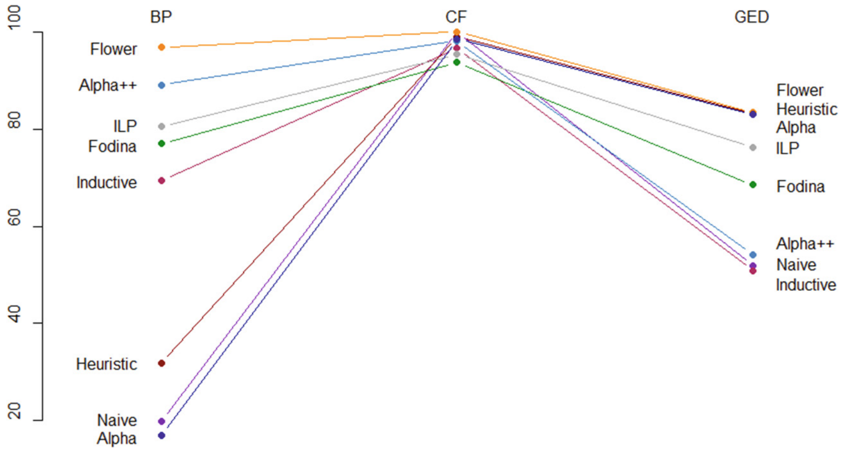

Fig. 5. Visualization of the average stability results when applying a resampling-based strategy. Averages over 20 artificial event logs.

The results can be found in Table 2. For a more intuitive representation of the results of the approaches only based on noise and only based on resampling, we refer to Figs. 4 and 5. Several observations can be made. First, it is clear that the results of the combined approach and the approach that only uses noise induction are highly similar. This was also touched upon in the previous section, and is likely due to the different impact of a 10 % noise induction compared to a 10 % resampling. Secondly, when comparing similarity metrics in Figs. 4 and 5, the ranges of the average stability appear to be similar across similarity metrics,

Table 2. Average and standard deviation of stability over 20 artificial and 5 real-life event logs, with a resampling and/or noise induction percentage of 10 %, where stability is being calculated using Behavioural Profiles (BP), Causal Footprints (CF) or Graph-edit Distance (GED).

Discovery technique	Perturbation strategy	Artificial stab(sd)			Real-life stab(sd)		
		BP	CF	GED	BP	CF	GED
Alpha	Resampling	0.17(0.27)	0.99(0.02)	0.83(0)	0.16(0.36)	0.86(0.05)	0.8(0.01)
Alpha++	Resampling	0.65(0.41)	0.99(0.02)	0.55(0.06)	0.76(0.15)	0.69(0.05)	0.62(0.09)
Flower	Resampling	0.98(0.1)	1(0)	0.83(0)	0.99(0.01)	0.99(0.01)	0.83(0)
Fodina	Resampling	0.84(0.2)	0.95(0.06)	0.69(0.05)	0.84(0.07)	0.9(0.04)	0.64(0.02)
Heuristic	Resampling	0.32(0.29)	0.99(0.02)	0.83(0)	0.72(0.16)	0.89(0.06)	0.78(0.04)
ILP	Resampling	0.5(0.49)	0.97(0.07)	0.77(0.09)	0.61(0.12)	0.62(0.18)	0.64(0.06)
Inductive	Resampling	0.7(0.15)	0.98(0.06)	0.51(0.06)	0.64(0.07)	0.91(0.1)	0.55(0.07)
Naive	Resampling	0.2(0)	1(0)	0.52(0.02)	0.21(0)	1(0)	0.53(0.05)
Alpha	Noise	0.04(0.06)	0.45(0.13)	0.56(0.05)	0(0)	0.35(0.19)	0.58(0.05)
Alpha++	Noise	0.05(0.08)	0(0.18)	0.39(0.05)	0.15(0.02)	−0.1(0.12)	0.42(0.04)
Flower	Noise	1(0)	1(0)	0.83(0)	0.83(0.05)	0.84(0.05)	0.83(0)
Fodina	Noise	0.52(0.3)	0.82(0.1)	0.65(0.06)	0.45(0.12)	0.68(0.12)	0.6(0.03)
Heuristic	Noise	0.3(0.27)	0.93(0.03)	0.81(0.01)	0.27(0.2)	0.59(0.17)	0.71(0.07)
ILP	Noise	0.14(0.23)	0.01(0.17)	0.51(0.02)	0.13(0.04)	−0.05(0.12)	0.56(0.05)
Inductive	Noise	0.24(0.08)	−0.03(0.13)	0.47(0.02)	0.35(0.1)	0(0.3)	0.51(0.02)
Naive	Noise	0.2(0.01)	1(0)	0.45(0.03)	0.2(0)	1(0)	0.51(0.04)
Alpha	Combined	0.03(0.06)	0.46(0.13)	0.56(0.05)	0(0)	0.33(0.17)	0.57(0.04)
Alpha++	Combined	0.08(0.08)	0(0.13)	0.38(0.05)	0.14(0.02)	−0.1(0.1)	0.42(0.04)
Flower	Combined	1(0)	1(0)	0.83(0)	0.82(0.04)	0.83(0.04)	0.83(0)
Fodina	Combined	0.53(0.29)	0.8(0.11)	0.64(0.06)	0.43(0.1)	0.65(0.11)	0.6(0.02)
Heuristic	Combined	0.3(0.27)	0.93(0.04)	0.81(0.01)	0.18(0.07)	0.57(0.17)	0.67(0.05)
ILP	Combined	0.13(0.19)	0.01(0.17)	0.51(0.02)	0.13(0.04)	−0.03(0.11)	0.57(0.04)
Inductive	Combined	0.24(0.07)	−0.02(0.13)	0.47(0.02)	0.34(0.11)	0.02(0.3)	0.51(0.02)
Naive	Combined	0.2(0.01)	1(0)	0.45(0.03)	0.2(0)	1(0)	0.51(0.04)

except for the combination of a resampling-based perturbation strategy with an underlying similarity metric based on causal footprints.

Regarding the different process discovery techniques, observe why we chose to incorporate flower miner and naive miner. A flower model is the least restrictive model one can imagine, where any activity can be executed in any order. It is clear that such a discovery technique should be very stable with regards to noise induction and resampling, and a very high stability is expected. Observe from Table 2 that this is indeed the case on the artificial logs. The inverse is true for the naive discovery technique, which naively incorporates any relationship between two activities seen in the log as long as they do not overlap in time. One would expect such a technique to score quite poorly on stability, which is the case when calculating its stability using behavioural profiles or Graph-edit Distance, but not using Causal Footprints. Apart from the results on the naive discovery technique, the results of the noise induction strategy are within expectations: techniques that were originally developed with robustness in mind, such as Heuristics miner and Fodina, achieve higher stability than techniques that are expected to be more sensitive to noise, such as ILP and Alpha.

Furthermore, Table 2 includes the standard deviation of the stability, which characterises the discrepancies across the 20 artificial and 5 real-life event logs, respectively. In general, the pure resampling-based perturbation strategy leads to more consistent results than the noise induction-approach. The stability based on GED appears to be the most consistent across event logs when combined with noise induction or a combined approach. When combined with resampling, the stability based on Causal Footprints and the stability based on GED perform equally consistent.

3.4 Effect of Log Characteristics

To show the effect of event log characteristics, we fit a number of regression models: a full and reduced one, for each perturbation strategy and each underlying similarity metric. All data is taken at a 10 %-level of perturbation. The full fitted models are of the form represented in Eq. 2, where x_{pd} are dummy variables indicating the process discovery technique and x_{dpi}, x_{ev}, x_{tl} are numerical variables representing the number of distinct process instances, number of distinct events and average trace length of the event log under scrutiny. In the restricted model (Eq. 3) the log characteristics are removed from the model. A lack-of-fit test is performed to show whether these log characteristics can be removed without substantial decrease in the fitness of the regression model. The results are represented in Table 3.

$$S^{pd} = \beta_0 + \sum_{pd=Alpha}^{Naive} \beta_{pd}x_{pd} + \beta_{dpi}x_{dpi} + \beta_{ev}x_{ev} + \beta_{tl}x_{tl} \tag{2}$$

$$S^{pd} = \beta_0 + \sum_{pd=Alpha}^{Naive} \beta_{pd}x_{pd} \tag{3}$$

Table 3. Fitness values and degrees of freedom for full and reduced linear model of stability over 20 artificial event logs, with a resampling or noise induction percentage of 10 %, with similarity being calculated using Behavioural Profiles (BP), Causal Footprints (CF) or Graph-edit Distance (GED). P values correspond to a chi-squared likelihood-ratio test on 3 degrees of freedom, and indicate whether the full model has a significantly higher goodness-of-fit than the reduced model.

Strategy	BP			CF			GED		
	R^2	LogLik	df	R^2	LogLik	df	R^2	LogLik	df
Noise: full	0.82	66	123	0.95	125.6	123	0.96	269.7	123
Noise: red	0.77	48.8	126	0.95	125	126	0.96	263.7	126
LRT: p-value			<**0.001**			0.80			**0.008**
	R^2	LogLik	df	R^2	LogLik	df	R^2	LogLik	df
Resampling: full	0.70	26	137	0.18	230	137	0.90	251	137
Resampling: red	0.65	13	140	0.15	226.8	140	0.90	248.8	140
LRT: p-value			<**0.001**			0.114			0.21

From Table 3, it is clear that excluding all log characteristics leads to a significant reduction in goodness-of-fit in the noise-induction case, except when combining this perturbation with causal footprints. This does not mean that all log characteristics are significantly related to the stability in all models: when combining noise induction with GED as similarity metric, only the average trace length is significantly related to the stability (p-value < 0.05), with a higher average trace length leading to lower stability. When combining noise induction or resampling with BP, the number of distinct activities is the only variable that is significant at a 5 %-level, with a higher number of activities leading to a lower expected stability, all other things equal.

4 Discussion and Future Work

Overview. The purpose of this paper is to propose a new dimension for the evaluation of process discovery techniques, stability, and a framework for its assessment. Specifically, a log-perturbation stability framework based on model similarity is extensively described, and its components are thoroughly evaluated using both artificial and real-life event logs. Two resampling strategies, one based on noise induction and one based on trace resampling, are proposed, and the strategy based on noise induction is shown to lead to the highest discriminating power, even at low percentages of noise induction. Three underlying similarity metrics are proposed, one based on behavioural profiles, one based on causal footprints, and one based on Graph-edit Distance.

Main Findings. Overall, there are three key takeaways from the experimental evaluation: (1) stability, as it was defined here, is a concept that can help differentiate between different process discovery techniques. (2) Different configurations of the stability framework lead to different results, however, making general conclusions about the process discovery techniques difficult. Both the type of perturbation and type of similarity metric influence these results. (3) The stability results are influenced by the specific characteristics of the event logs used, such as number of activities in this log, or the average trace length, apart from the configurations with Causal Footprints.

Limitations. Several limitations exist with regard to our framework and its evaluation, and these limitations should be addressed more thoroughly in future work. First of all, there is the question of the desirability of certain types of perturbation, especially noise induction. Although noise induction is shown to lead to interesting results, it may not be an accurate representation of reality, given that most information systems record events rather faithfully nowadays. Secondly, a limitation exists regarding the evaluation of discovery algorithms: since our approach requires a number of iterations in order to create valid results, we disregarded discovery algorithms that don't scale well (e.g. the genetic approaches of [3,6]). Moreover, it should be noted that the results presented here are valid only for the distinct configurations of the algorithms. Clearly, the results would be different for Inductive miner, for example, given that a more inclusive or exclusive configuration is be used. Therefore, 'parameter stability' and 'log perturbation stability' are interrelated concepts. Nonetheless, applying an evaluation on the default settings of an algorithm is justifiable, given that users of these implementations rarely change the default settings [4]. Thirdly, the choice of Petri nets as an underlying construct for discovered process models leads to an inherent bias, especially with regards to the conversion of techniques that produce results in an other modelling notation. Fourthly, the current implementation is limited to existing process similarity metrics. However, these metrics were not conceived with discovered process models in mind. Therefore, **future work** could look into the construction of a process similarity metric geared specifically towards stability of process discovery techniques. An alternative to this approach would be to quantify similarity in a more event-log driven way, by incorporating conformance checking metrics into the similarity. An example of this could be behavioural precision and recall as defined in [19].

Finally, two relationships should be quantified in continuing work: on the one hand, the relationship between the stability results and specific behavioural structures present in event logs, or the models used to generate these event logs, such as loops of a certain length and invisible transitions; on the other hand, the relationships between the proposed approach and related conformance checking metrics for quality metrics like cross-validation-based generalization [20].

References

1. Bae, J., Caverlee, J., Liu, L., Yan, H.: Process mining by measuring process block similarity. In: Eder, J., Dustdar, S. (eds.) BPM Workshops 2006. LNCS, vol. 4103, pp. 141–152. Springer, Heidelberg (2006)
2. Bolt, A., de Leoni, M., van der Aalst, W.M.P.: Scientific workflows for process mining: building blocks, scenarios, and implementation. Int. J. Softw. Tools Technol. Transf. 1–22 (2015). doi:10.1007/s10009-015-0399-5
3. Borja, V., Mucientes, M., Lama, M.: ProDiGen: mining complete, precise and minimal structure process models with a genetic algorithm. Inf. Sci. Innov. Appl. Artif. Neural Netw. Eng. **294**, 315–333 (2015)
4. Claes, J., Poels, G.: Process mining and the ProM framework: an exploratory survey. In: La Rosa, M., Soffer, P. (eds.) BPM Workshops 2012. LNBIP, vol. 132, pp. 187–198. Springer, Heidelberg (2013)
5. Davison, A.C., Hinkley, D.V.: Bootstrap Methods and Their Application, vol. 1. Cambridge University Press, Cambridge (1997)
6. De Medeiros, A.K.A., Weijters, A.J.M.M., Van Der Aalst, W.M.P.: Genetic process mining: an experimental evaluation. Data Min. Knowl. Discov. **14**(2), 245–304 (2007)
7. De Weerdt, J., De Backer, M., Vanthienen, J., Baesens, B.: A multi-dimensional quality assessment of state-of-the-art process discovery algorithms using real-life event logs. Inf. Syst. **37**(7), 654–676 (2012)
8. Dijkman, R., Dumas, M., Van Dongen, B., Krik, R., Mendling, J.: Similarity of business process models: metrics and evaluation. Inf. Syst. **36**(2), 498–516 (2011)
9. van Dongen, B.F., Dijkman, R.M., Mendling, J.: Measuring similarity between business process models. Adv. Inf. Syst. Eng. **5074**, 450–464 (2008)
10. van Dongen, B.F., Mendling, J., van der Aalst, W.M.P.: Structural patterns for soundness of business process models. In: 10th IEEE International Enterprise Distributed Object Computing Conference, pp. 116–128 (2006)

11. Goedertier, S., Martens, D., Vanthienen, J., Baesens, B.: Robust process discovery with artificial negative events. J. Mach. Learn. Res. **10**, 1305–1340 (2009)
12. Hastie, T., Tibshirani, R., Friedman, J.: Unsupervised learning. In: Hastie, T., Tibshirani, R., Friedman, J. (eds.) The Elements of Statistical Learning: Data Mining, Inference Prediction, 2nd edn, pp. 485–585. Springer, New York (2009)
13. Kunze, M., Weidlich, M., Weske, M.: Behavioral similarity – a proper metric. In: Rinderle-Ma, S., Toumani, F., Wolf, K. (eds.) BPM 2011. LNCS, vol. 6896, pp. 166–181. Springer, Heidelberg (2011)
14. Lange, T., Roth, V., Braun, M.L., Buhmann, J.M.: Stability-based validation of clustering solutions. Neural Comput. **16**(6), 1299–1323 (2004)
15. Lee, Y., Lee, J., Jun, C.H.: Stability-based validation of bicluster solutions. Pattern Recogn. **44**(2), 252–264 (2011)
16. Leemans, S.J.J., Fahland, D., van der Aalst, W.M.P.: Discovering block-structured process models from event logs - a constructive approach. In: Colom, J.-M., Desel, J. (eds.) PETRI NETS 2013. LNCS, vol. 7927, pp. 311–329. Springer, Heidelberg (2013)
17. Levine, E., Domany, E.: Resampling method for unsupervised estimation of cluster validity. Neural Comput. **13**(11), 2573–2593 (2001)
18. Maruster, L.: A Machine Learning Approach To Understand Business Processes. Eindhoven University of Technology, Eindhoven (2003)
19. de Medeiros, A.K.A., van der Aalst, W.M.P., Weijters, A.J.M.M.: Quantifying process equivalence based on observed behavior. Data Knowl. Eng. **64**(1), 55–74 (2008)
20. Van der Aalst, W., Adriansyah, A., Van Dongen, B.: Replaying history on process models for conformance checking and performance analysis. Wiley Interdiscip. Rev. Data Min. Knowl. Discov. **2**(2), 182–192 (2012)
21. Van Der Aalst, W., Weijters, T., Maruster, L.: Workflow mining: discovering process models from event logs. IEEE Trans. Knowl. Data Eng. **16**(9), 1128–1142 (2004)
22. Vanden Broucke, S.K.L.M.: Artificial negative events and other techniques. Ph.D. thesis, KU Leuven (2014)
23. Vanden Broucke, S.K.L.M., De Weerdt, J., Vanthienen, J., Baesens, B.: Determining process model precision and generalization with weighted artificial negative events. IEEE Trans. Knowl. Data Eng. **26**(8), 1877–1889 (2014)
24. Weidlich, M., Polyvyanyy, A., Desai, N., Mendling, J., Weske, M.: Process compliance analysis based on behavioural profiles. Inf. Syst. **36**(7), 1009–1025 (2011)
25. Weidlich, M., Polyvyanyy, A., Mendling, J., Weske, M.: Efficient computation of causal behavioural profiles using structural decomposition. In: Lilius, J., Penczek, W. (eds.) PETRI NETS 2010. LNCS, vol. 6128, pp. 63–83. Springer, Heidelberg (2010)
26. Weijters, A.J.M.M., van der Aalst, W.: Rediscovering workflow models from event-based data using little thumb. Integr. Comput. Eng. **10**, 151–162 (2003)
27. Wen, L., Van Der Aalst, W.M.P., Wang, J., Sun, J.: Mining process models with non-free-choice constructs. Data Min. Knowl. Discov. **15**(2), 145–180 (2007)
28. van der Werf, J.M.E.M., van Dongen, B.F., Hurkens, C.A.J., Serebrenik, A.: Process discovery using integer linear programming. In: van Hee, K.M., Valk, R. (eds.) PETRI NETS 2008. LNCS, vol. 5062, pp. 368–387. Springer, Heidelberg (2008)
29. Zha, H., Wang, J., Wen, L., Wang, C., Sun, J.: A workflow net similarity measure based on transition adjacency relations. Comput. Ind. **61**(5), 463–471 (2010)