HOW TO EVALUATE PROCESS DISCOVERY FOR DIGITAL TWINS IN INDUSTRY 4.0? PROCESS DISCOVERY, HYPOTHESIS TESTING, AND CONFORMANCE ANALYSIS

Juliano Yoshiro[1], Paulo Victor Lopes[2], Filipe Alves Neto Verri[3], and Anders Skoogh[4]

[1] Dept. of Computer Methods, Instituto Tecnológico de Aeronáutica: juliano.nishiura@ga.ita.br
[2] Dept. of Industrial and Materials Sci., Chalmers University of Tech.: paulov@chalmers.se
[2] Dept. of Computer Methods, Instituto Tecnológico de Aeronáutica: paulo.lopes@ga.ita.br
[3] Dept. of Computer Methods, Instituto Tecnológico de Aeronáutica: filipe.verri@gp.ita.br
[4] Dept. of Industrial and Materials Sci., Chalmers University of Tech.: anders.skoogh@chalmers.se

KEYWORDS

Production Lines, Conformance Analysis, Hypothesis Testing, Bootstrap Simulation.

ABSTRACT

The field of data science is an emerging area of study that arises in the context of the production of a large volume of data in recent years. The objective of this area is to obtain valuable information that is extracted through data processing. In the industrial context, the identification of failures and bottlenecks in production lines is essential to increase the productivity of the evaluated systems. However, manual analysis can be time-consuming and costly. Process discovery is a set of techniques that includes the use of algorithms to extract a process model from the event log, which can be used as a basis for developing Digital Twins. Therefore, this paper proposes the use of an artificial production line generator so that process mining algorithms can be tested with a large number of samples and different network characteristics. Thus, the main contribution will be the testing of hypotheses to assist in choosing the best algorithms in a practical context.

INTRODUCTION

The field of data science is an emerging area of study that arises within the context of the production of a large volume of data in recent years. The goal of this area is to obtain valuable information that is extracted through data processing, which can be achieved by:

1) Visualization, by selecting, filtering, and organizing data into graphs in order to gain insights;
2) Statistical approach, by applying statistical models and making hypothesis formulations;
3) Predictive model building, applying machine learning tools so that the model can learn from the data and identify patterns that are not easily perceptible by data visualization.

It is worth noting that the field of data science is not limited to just a specific area of knowledge. In fact, there are many areas where there is a considerable amount of data production, such as healthcare, industry, the public and private sectors, commerce, and financial markets. Thus, a data scientist may approach data in a different way according to the needs of the area and the subject matter knowledge involved during the work.

Process Science, on the other hand, is a discipline that combines knowledge of information technology with business management science, with the goal of optimizing and improving processes [1]. With the advent of data science and the approach of extracting knowledge from data, the science of process mining emerges, which is defined as an area of study concerned with extracting a model from a data repository, known as event logs, generated by the real process [2], and analyzing the extracted model under the metrics of conformance analysis for further process enhancement. The motivation behind this field of study lies in the interest in tracking and observing real process behavior and identifying errors that traditional process science tools are not able to detect [3].

Currently, Process Mining is divided into 4 main sub-areas: process discovery, process conformance, process enhancement, and minor areas [2]. Process discovery is a set of techniques that includes the use of algorithms for the extraction of a process model from the event log. Process conformance or conformance analysis is a step that compares the extracted model with the behavior contained in the event log to verify adequacy [4]. It can also be used to check the robustness of the models generated by the discovery algorithm [1]. Process enhancement is a further step that involves improving and enhancing the model by adding more extensions or more data. Although Process Mining is growing as a study interest, the field faces problems concerning its methods and reproducibility.

In the part of conformance analysis, which is a recent sub-area within Process Mining, there have been efforts to systematize the metrics for process conformance, as the work of Roozinat and Aalst [5] represents. However, there is still no consolidated consensus on the application of analysis metrics, and

Communications of the ECMS, Volume 38, Issue 1,
Proceedings, ©ECMS Daniel Grzonka, Natalia Rylko,
Grazyna Suchacka, Vladimir Mityushev (Editors) 2024
ISBN: 978-3-937436-84-5/978-3-937436-83-8(CD) ISSN 2522-2414

new evaluation metrics are developed periodically to better validate the performance of process discovery algorithms.

Another obstacle to conducting research is the availability and provision of data for applying new techniques in the field of study. This is due to the sensitivity of the data provided by entities, the security issues that involve companies, and the privacy of those who generate the data. These factors together contribute to difficulty in accessing databases with a good quality of data [6]. A solution proposed to solve this problem is the CLEMATIS algorithm [6], which is an artificial data generator of a simulated industrial production line. It can also generate the Petri-net of the simulated artificial process through process discovery besides the data in the form of event logs.

The purpose of this work is to provide a method, using hypothesis testing and bootstrapping simulation, to study the behavior of process discovery algorithms when faced with different types of production networks. This work also intends to validate the functionality of CLEMATIS as a reliable artificial event log generator by analyzing the values returned with conformance checking, as well as to provide a reference value, each corresponding to a type of production line, miner algorithm, and evaluation metric.

THEORETICAL REFERENCE

There are currently four main conformance checking metrics that compare an extracted model to its event log: fitness, precision, simplicity, and generalization. In other studies, precision is referred to as one of the conformance checking dimensions that contain the behavioral appropriateness metric, and simplicity is referred to as structural appropriateness [5]. It is also pertinent to say that there are different methods for evaluating each metric, as illustrated in the work of Blum [7].

The four parts of this section focus solely on techniques that were actually applied in this work and will be dedicated to showing how each of these metrics can be calculated.

Fitness

The fitness metric evaluates the model's accuracy in representing all the traces contained in the event log [5]. The most common method employed to calculate its value is token-based replay [1].

The example in Figure 1 shown in this subsection is a Petri-net model extracted by Alpha Miner on the event log of a short line network with 4 steps simulated by CLEMATIS. The frequency of each trace is shown in Table 1.

For the trace labeled as "7486", Figures 2, 3, 4, and 5 illustrate in detail the process of counting the following parameters that are important to measure fitness:

- m : for amount of missing tokens.
- r : for amount of remaining tokens.
- c : for amount of consumed tokens.
- p : for amount of produced tokens.

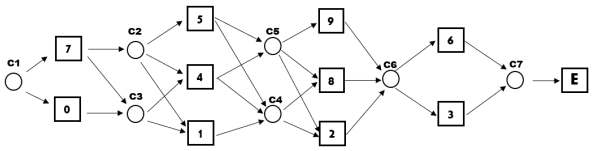

Figure 1. Discovered process model example for conformance evaluation.

Traces	Frequency	Traces	Frequency
0123	32	7423	2
7486	25	7596	1
0126	9	7583	1
7483	7	0426	1
0186	5	7126	1
7496	4	7493	1
7123	4	7526	1
7593	3	0493	1
0486	2	—	—

TABLE 1. EXAMPLE OF TRACES WITH COUNTED FREQUENCY.

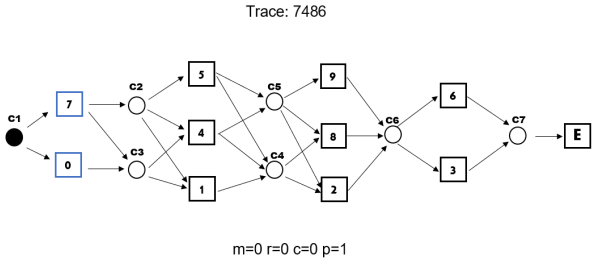

Figure 2. In first step, a token is produced in "C1", enabling acitivities '0' and '7'.

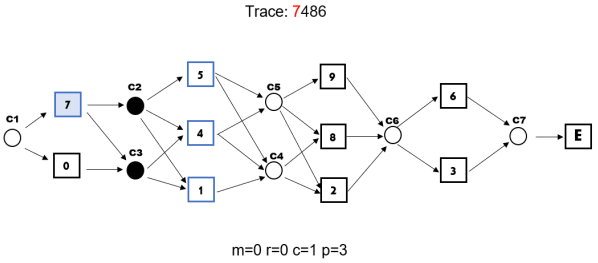

Figure 3. The token in "C1" is consumed to conduct the activity labeled as "7". This activity places tokens in both "C2" and "C3", enabling '5', '1' and '4'.

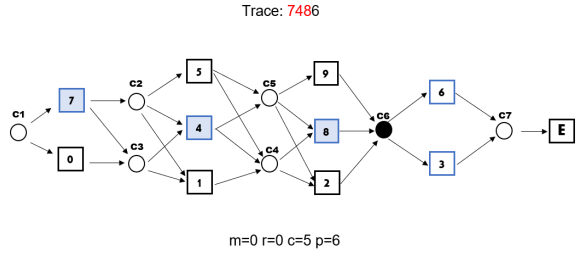

Figure 4. After completing the task "4", activity "8" is done and the tokens that were in "C4" and "C5" are consumed.

The formula for measuring fitness is:

$$f = \frac{1}{2}\left(1 - \frac{\sum_{i=1} n_i m_i}{\sum_{i=1} n_i c_i}\right) + \frac{1}{2}\left(1 - \frac{\sum_{i=1} n_i r_i}{\sum_{i=1} n_i p_i}\right) \quad (1)$$

where i represent the index of the trace and n_i is

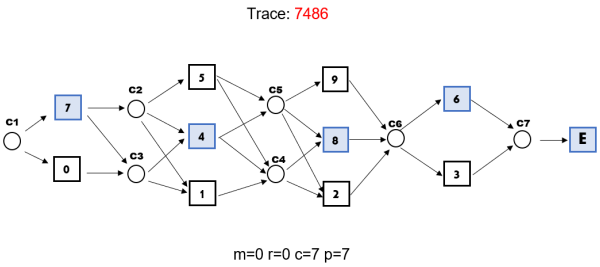

Figure 5. The trace is completed once the token located in "C7" is consumed to end the process.

the number of occurrences of the i-th trace in event log [1].

Precision

It is desirable to obtain a balanced model in terms of underfitting. For this purpose, the precision metric evaluates the model to check whether it is reproducing non-existent behavior in the event log [2].

For precision, it is worth mentioning that this term is also referred to as a quality dimension, and within this dimension, there are several metrics to assess it. In this work, we will be referring to ETC-precision as one of these metrics, and it will be discussed as a different approach for determining the precision metric.

The need for the precision metric arises when the model predicts not only the correct traces but also traces that do not occur in the original log. In the model, this could be represented as additional existing branches in the Petri-net. In this case, the model is too generic and uninformative, and it would require effort to identify paths that actually represent the real process.

ETC-precision relies on the concept of *escaping edges*, which is the number of allowed task transitions subtracted by number of reflected task transitions on a given state. For clarification, let each step shown in Figures 2, 3, 4, and 5 be an example of states. The total number of escaping edges at the state described by the Figure 4 is 0 because in this state, tasks '3' and '6' are enabled, but task '6' is expected by the trace "7486" and task '3' is verified after the sequence "748" in the trace "7483". Since both traces are present in the Table 1, the total number of escaping edges for this particular state is $2 - 2 = 0$. The application of the method on every state of the trace "7486" returns the total number of escaping edges, which is 0 in this example. After the explanation of the concept of escaping edges, ETC-precision can be calculated by [4]:

$$etc_P = 1 - \frac{\sum_{i=1} \sum_{j=1} e_{ij}}{\sum_{i=1} \sum_{j=1} a_{ij}} \tag{2}$$

where e_{ij} represent the number of escaping edges for i-th trace in j-th state and a_{ij} represent the number of enabled transitions for i-th trace in j-th state.

The example of the trace "7486", covered in this work, returns $etc_P \approx 0.9928$.

Generalization

In contrast, the generalization metric is concerned with the overfitting of the model and verifies whether it is representing the real process or only the traces contained in the event log that generated it [8]. Another suitable definition for generalization is to verify whether the model can represent future behaviors of the real process or an unknown system [9]. It is calculated in terms of the frequency of activities and the number of nodes [10]:

$$Q_G = 1 - \frac{\sum_i \frac{1}{\sqrt{(n_i)}}}{N} \tag{3}$$

where n_i represents the number of occurrences of activity i in the event log and N represents the total number of nodes in Petri-net model. The reasoning behind the formula assumes that the least visited nodes in graph (least performed activities) could represent a bad generalization from the model [10]. Hence, if n_i were low for some i-th activity, then the subtracting term in equation (3) would increase and the value of parameter Q_G would decrease. It is important to remark that this method does not consider invisible tasks.

Simplicity

Simplicity is a metric designed to evaluate the degree of complexity of the mined process by analyzing the structure of the model. The criteria to calculate this metric take into account the weighted average arc degree per node [11]:

$$x_m = \frac{outgoing\ arc + incoming\ arc}{number\ of\ nodes} \tag{4}$$

The method for calculating the simplicity is [7]:

$$S = \frac{1}{1 + max\{x_m - x_r, 0\}} \tag{5}$$

Here x_r refers to the weighted average arc degree of the real model. It is important to note that, in this formula, nodes actually represent activities that occur in the model and not the token places. It also allows us to state that the higher the value calculated for simplicity, the less complex the discovered model is.

METHODOLOGY

The research problem is divided into 4 steps. In the first step, the artificial data generator CLEMATIS is used to simulate 4 basic types of production networks, providing data in .XES format. At the end of this step, 4000 examples are generated, where each .XES file corresponds to one example, and each type of production network has 1000 examples.

In the following step, Process Discovery algorithms are used to extract a Petri-net model from each example. The chosen algorithms are the alpha miner, inductive miner, and heuristics miner from the library PM4Py. At the end of this step, 12000 Petri-net models are generated, with each set of 4000 models corresponding to a specific algorithm.

In the third step, the conformance checking metrics are evaluated for each set of 1000 Petri-net models. Since in this work 4 evaluation metrics will be used, at the end of this step there will be 48000 metric values, each 1000 corresponding to a specific production network, algorithm and, conformance metric.

In the final step, each set of 1000 examples is treated as the original sample for Bootstrapping simulation [12] with hypothesis testing. A summary of the proposed work is shown in the flowchart below.

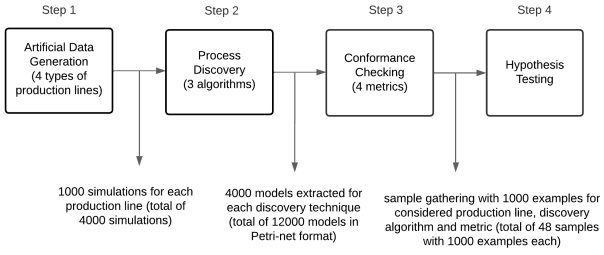

Figure 6. Flowchart of the work.

Artificial Event Log

In this subsection, the 4 types of production networks and the method used to generate them are explained and detailed. The event logs used in this work are all derived from the CLEMATIS algorithm, which generates simulated production networks by taking n, the number of machines used in process, and s, the number of production steps, as inputs. Then, the α parameter is defined as follows [6]:

$$\alpha = \frac{s}{n} \tag{6}$$

The parameters s and α are crucial to determine the character of the simulated production process. For $\alpha = 1$ as example, it results in a production line that shows only one operating machine for each production step. Since s can not be greater than n (there would be more production steps than operating machines, which is incoherent), the α parameter is upwardly constrained by 1. Hence, its value is ranging from 0 to 1. For smaller values of α, the resulting production line becomes wider. For $\alpha = 0.5$, there are in average two functioning machines per production step. For values of α less than 0.5, the production line starts to become denser.

With the understanding that s and α parameters regulates the length and the density of the artificial production line respectively, there are 4 types of production network generated by CLEMATIS:

- Short and non-dense lines (SND).
- Short and dense lines (SD).
- Large and non-dense lines (LND).
- Large and dense lines (LD).

In this present work, we will regard production line with $0.15 \leq \alpha \leq 0.3$ as dense and the interval $0.5 \leq \alpha \leq 1$ as non-dense. For s parameter, we define the value contained in interval between 5 and 12 as belonging to short line group and the interval 18 and 25 as belonging to large line group.

After each simulation done by the CLEMATIS algorithm, the data regarding this simulation are stored in .txt format, which is further converted into .XES, the appropriate extension for working with Process Discovery and Conformance Checking using the PM4Py library.

Process Discovery and Conformance Checking Implementation

The tools used to discover processes from the event log and to evaluate the model from the perspective of 4 basic conformance metrics cited in section are from the PM4Py implementation. PM4Py is an open-source library that provides a diversity of tools for Process Mining with increasing use and acceptance within industrial and academic fields [13]. The Alpha Miner, Inductive Miner, and Heuristics Miner implemented in Python [14] are used for Process Discovery steps in this work. Since the tools used for this current work are already available, it is necessary to review their algorithmic implementation to verify the possible implications in the methodology proposal.

The implementation of Alpha Miner or Mining Algorithm α is derived from Van der Aalst, Weijters and Măruşter work [15]. The algorithm assumes that two tasks are connected if a causality can be discovered through the event log inspection. This approach works well for SWF-nets (Structured Workflow Nets), which are workflow-nets defined by the article as not containing transitions with free-choice and synchronization at the same time. However, its limitation includes the difficulty of rediscovering SWF-nets with short-loop inside the flow, which is not a problem, since CLEMATIS is an event log generator that simulates an unidirectional production network. Another important remark regarding this algorithm is its complexity, which is linear to the size of the event log when building links between tasks and exponential to the amount of the identified tasks in the following steps [15].

Inductive Miner algorithm is derived from the work of Van der Aalst, Fahland and Leemans [16]. It is based in cut detection, splitting of directly-follow graph into smaller ones to make a recursion until base cases are discovered. In this approach, a Petri-net is viewed as process tree. Its limitations include difficulty in extracting a good model in terms of fitness, precision, simplicity and generalization for event logs with infrequency, incompleteness and noise. In the case of incompleteness, which is defined as absence or low quantity of examples of traces that might appear in the event log [16], the proposed solution is to let the mining algorithm guess the most probable cut for that trace.

Heuristics Miner is an approach that derives from dependency graph [17] and also relies on causality between two tasks. Its robustness to noise and its response to less common behavior, which is not properly addressed by a common Inductive Miner.

For Conformance Checking, all the methods described in previous section are in agreement with the tools provided by PM4Py. For instance, fitness and precision will be calculated through token-based

replay [14] and generalization will be calculated according to the equation (3).

Hypothesis Testing and Bootstrapping Simulation

For hypothesis testing, the Type 1 error was adopted. For each sample containing 1000 values for a specific metric, an average value was assumed as the null hypothesis, and a significance level of β = 0.01 was adopted, which means that if a p-value not greater than 0.01 is achieved, then we can reject the null hypothesis with 99 % of confidence. Since a sample with only 1000 is considerably small for hypothesis testing, a Bootstrapping simulation [12] was performed with 10,000 resampling using half the quantity of examples from the original sample. For each example in the resampling, a corresponding test statistic was calculated according to the formula:

$$E = \frac{\hat{X} - \mu}{\sigma} \tag{7}$$

where E is the statistic, $\hat{X}$ is the mean of the resampling, μ is the assumed sample mean under the null hypothesis, and $\sigma$is the experimental standard deviation. For the test statistic of the original sample, the following equation is used instead:

$$E = \frac{\hat{X} - \mu}{\frac{\sigma}{\sqrt{n}}} \tag{8}$$

where all the variables have the same meaning as in Equation 7 except for the variable n, which represents the number of examples in the original sample. For hypothesis testing, a significance level of β = 0.01 was adopted, and the p-value was calculated using the following formula:

$$P = \frac{Q}{N} \tag{9}$$

Where p is the p-value, Q is the number of cases where the test statistic of the original sample has a value greater than the test statistic of the resampling, and N is the total number of resampling performed. If the calculated p-value is not inferior to the significance level, then a new lower average is assumed as the new null hypothesis, and the described process is repeated until a p-value less than 0.01 is achieved. Once the condition is met, then it is possible to assume that the average value of the original sample is not lower than the average value assumed for the null hypothesis.

RESULTS AND DISCUSSION

Hypothesis testing and Bootstrapping simulation were performed on 48 samples, giving 48 values derived from the negated null hypothesis, which will be called in this section as lower limits. These values are displayed in two separated tables, where the Table 2 contains the lower limits for fitness and precision metrics and the Table 3 contains the lower limits for simplicity and generalization. For comparison, the average value for 48 samples were calculated and displayed besides the lower limits between parenthesis. The lower limit express a value that is the least that can be expected for the original sample with 99 % of confidence, if the same experimental conditions of this work are reproduced.

TABLE 2. LOWER BOUNDS OBTAINED THROUGH HYPOTHESIS TESTING FOR THE 24 SAMPLES (FITNESS AND PRECISION). THE ORIGINAL SAMPLE MEANS ARE IN PARENTHESES, AND THE VALUES IN BOLD REPRESENT THE HIGHEST LOWER BOUND OBTAINED FOR A METRIC FOR A GIVEN PRODUCTION LINE.

	Algorithm	Fitness	Precision
SND	Alpha Miner	0,98 (0,99)	**0,99** (0,99)
	Inductive Miner	(*) (1,00)	0,94 (0,96)
	Heuristics Miner	**0,99** (0,99)	0,90 (0,94)
SD	Alpha Miner	0,78 (0,81)	**0,97** (0,98)
	Inductive Miner	(*) (1,00)	0,24 (0,31)
	Heuristics Miner	**0,88** (0,90)	0,69 (0,75)
LND	Alpha Miner	0,98 (0,98)	**0,97** (0,97)
	Inductive Miner	(*) (1,00)	0,90 (0,92)
	Heuristics Miner	**0,99** (0,99)	0,86 (0,91)
LD	Alpha Miner	0,76 (0,78)	**0,97** (0,98)
	Inductive Miner	(*) (0,99)	0,18 (0,23)
	Heuristics Miner	**0,89** (0,91)	0,61 (0,68)

TABLE 3. LOWER BOUNDS OBTAINED THROUGH HYPOTHESIS TESTING FOR THE 24 SAMPLES (SIMPLICITY AND GENERALIZATION). THE ORIGINAL SAMPLE MEANS ARE IN PARENTHESES, AND THE VALUES IN BOLD REPRESENT THE HIGHEST LOWER BOUND OBTAINED FOR A METRIC FOR A GIVEN PRODUCTION LINE.

	Algorithm	Simplicity	Generalization
SND	Alpha Miner	0,81 (0,85)	0,82 (0,84)
	Inductive Miner	0,82 (0,86)	0,82 (0,83)
	Heuristics Miner	**0,90** (0,93)	**0,84** (0,86)
SD	Alpha Miner	0,30 (0,33)	0,73 (0,75)
	Inductive Miner	0,47 (0,50)	0,75 (0,77)
	Heuristics Miner	**0,82** (0,85)	**0,78** (0,79)
LND	Alpha Miner	0,78 (0,82)	0,83 (0,84)
	Inductive Miner	0,79 (0,82)	0,83 (0,84)
	Heuristics Miner	**0,88** (0,90)	**0,85** (0,86)
LD	Alpha Miner	0,31 (0,34)	0,72 (0,73)
	Inductive Miner	0,47 (0,50)	0,74 (0,75)
	Heuristics Miner	**0,84** (0,86)	**0,79** (0,80)

Firstly, it's noticed that the lower limits established through the hypothesis test for a significance level β = 0.01 are near to the values of the original sample means, which attests to the consistency of the values obtained for the 4 metrics in each discovery algorithm. By examination of performances returned by algorithms, it's observed from the lower thresholds that the heuristics miner technique manages to extract models with the highest values for the simplicity and generalization metrics, while the alpha miner extracts better models in terms of precision metric. Despite the sample mean fitness values for the inductive miner being close to or equal to 1, the technique presented a lower limit value for precision in dense networks. It is necessary to remark, however, that it was not possible to calculate the values for lower limits of inductive miner on metric fitness. This is explained by the standard deviation of the original samples, which were approximately zero, although their average scores were almost equal to 1. Since the hypothesis testing relies on a non null standard deviation to be applied, it was decided to omit their calculation.

On general terms, it is interesting to note that the performance of the Process Discovery algorithms are not directly affected by the length of the production network as the two tables show. In short and large dense production networks, for example, the algorithms presents similar performance in the 4 evaluated metrics. By comparing two production networks with different density, however, it is noticeable that the evaluated metrics are lower for denser production lines. It indicates that the complexity rather than length of a production network interferes more on performance of Process Discovery algorithms.

CONCLUSIONS AND FUTURE WORK

We concluded that each algorithm showed a different performance according to the structure of the evaluated networks. The tables can assist in expanding the use of these algorithms in a more elucidated manner, as well as to serve as reference values for performance of each process discovery algorithm. This work, can be extended by studying the behaviour of process discovery algorithms on production networks with continuously increasing density to better understand the relationship between performance and complexity. Besides that, in this study only the techniques and evaluation methods provided by PM4Py library were applied, which can be improved in the future by applying more techniques and different conformance checking metrics to generate values that can be used as references.

ACKNOWLEDGMENTS

I would like to thank the National Council for Scientific and Technological Development (CNPq), Technological Institute of Aeronautics (ITA), Coordination for the Improvement of Higher Education Personnel (CAPES), and Chalmers University of Technology for the support and funding provided.

References

[1] Wil Van Der Aalst. *Process mining: data science in action*, volume 2. Springer, 2016.

[2] Cleiton dos Santos Garcia, Alex Meincheim, Elio Ribeiro Faria Junior, Marcelo Rosano Dallagassa, Denise Maria Vecino Sato, Deborah Ribeiro Carvalho, Eduardo Alves Portela Santos, and Edson Emilio Scalabrin. Process mining techniques and applications–a systematic mapping study. *Expert Systems with Applications*, 133:260–295, 2019.

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

[4] Jorge Munoz-Gama and Josep Carmona. A fresh look at precision in process conformance. In *Business Process Management: 8th International Conference, BPM 2010, Hoboken, NJ, USA, September 13-16, 2010. Proceedings 8*, pages 211–226. Springer, 2010.

[5] Anne Rozinat and Wil MP Van der Aalst. Conformance checking of processes based on monitoring real behavior. *Information Systems*, 33(1):64–95, 2008.

[6] Paulo Victor Lopes, Leonardo Silveira, Roberto Douglas Guimaraes Aquino, Carlos Henrique Ribeiro, Anders Skoogh, and Filipe Alves Neto Verri. Synthetic data generation for digital twins: enabling production systems analysis in the absence of data. *International Journal of Computer Integrated Manufacturing*, pages 1–18, 2024.

[7] F Rojas Blum. Metrics in process discovery. *Tech. Rep. Technical Report TR/DCC-2015–6, Computer Science Dept., University of Chile*, 2015.

[8] Sebastian Dunzer, Matthias Stierle, Martin Matzner, and Stephan Baier. Conformance checking: a state-of-the-art literature review. In *Proceedings of the 11th international conference on subject-oriented business process management*, pages 1–10, 2019.

[9] Joos CAM Buijs, Boudewijn F van Dongen, and Wil MP van der Aalst. Quality dimensions in process discovery: The importance of fitness, precision, generalization and simplicity. *International Journal of Cooperative Information Systems*, 23(01):1440001, 2014.

[10] Joos CAM Buijs, Boudewijn F Van Dongen, Wil MP van Der Aalst, et al. On the role of fitness, precision, generalization and simplicity in process discovery. In *OTM Conferences (1)*, volume 7565, pages 305–322, 2012.

[11] Laura Sánchez-González, Félix García, Jan Mendling, Francisco Ruiz, and Mario Piattini. Prediction of business process model quality based on structural metrics. In *Conceptual Modeling–ER 2010: 29th International Conference on Conceptual Modeling, Vancouver, BC, Canada, November 1-4, 2010. Proceedings 29*, pages 458–463. Springer, 2010.

[12] Patrick O'Keefe and Joseph Lee Rodgers. A simulation study of bootstrap approaches to estimate confidence intervals in defries–fulker regression models (with application to the heritability of bmi changes in the nlsy). *Behavior genetics*, 50(2):127–138, 2020.

[13] Alessandro Berti, Sebastiaan van Zelst, and Daniel Schuster. Pm4py: a process mining library for python. *Software Impacts*, 17:100556, 2023.

[14] Alessandro Berti, Sebastiaan J Van Zelst, and Wil van der Aalst. Process mining for python (pm4py): bridging the gap between process-and data science. *arXiv preprint arXiv:1905.06169*, 2019.

[15] Wil Van der Aalst, Ton Weijters, and Laura Maruster. Workflow mining: Discovering process models from event logs. *IEEE transactions on knowledge and data engineering*, 16(9):1128–1142, 2004.

[16] Sander JJ Leemans, Dirk Fahland, and Wil MP Van der Aalst. Scalable process discovery and conformance checking. *Software & Systems Modeling*, 17:599–631, 2018.

[17] AJMM Weijters and Joel Tiago S Ribeiro. Flexible heuristics miner (fhm). In *2011 IEEE symposium on computational intelligence and data mining (CIDM)*, pages 310–317. IEEE, 2011.

AUTHOR BIOGRAPHIES

JULIANO YOSHIRO is currently a bachelor's student in Computer Engineering at the Computer Science Division, Instituto Tecnológico de Aeronáutica (ITA). His research interests include process mining, conformance analysis, statistical analysis, data science and machine learning. His email address is juliano.nishiura@ga.ita.br.

PAULO VICTOR LOPES is a Research Assistant at the Industrial and Material Science Department of Chalmers University of Technology and Ph.D. student in Operations Research at Instituto Tecnológico de Aeronáutica and Universidade Federal de São Paulo. His research interests include digital twins, traceability systems, process mining, digital platforms, data-driven decision making and production systems. His e-mail address is paulov@chalmers.se.

FILIPE VERRI is currently an Assistant Professor with the Division of Computer Science,

Department of Computer Methods, Instituto Tecnológico de Aeronáutica (ITA). His research interests include data science, machine learning, complex networks, and complex systems. He is also a founding member of the DroneComp Research Group, that focuses on developing computer systems and computational methods for the next-generation air transportation system. Before starting his Professorship career, he graduated summa cum laude in bachelor and Ph.D. from the University of São Paulo. His email address is verri@ita.br.

ANDERS SKOOGH is a Professor at Industrial and Material Science, Chalmers University of Technology. He is a research group leader for Production Service & Maintenance Systems. Anders is also the director of Chalmers' Masters' program in Production Engineering and board member of the think-tank Sustainability Circle. Before starting his research career, he accumulated industrial experience from being a logistics developer at Volvo Cars. His email address is anders.skoogh@chalmers.se.