Fig4PM: A Library for Calculating Event Log Measures (Extended Abstract)

Fareed Zandkarimi
University of Mannheim
Mannheim, Germany
zandkarimi@uni-mannheim.de

Peter Decker
University of Mannheim
Mannheim, Germany

Jana-Rebecca Rehse
University of Mannheim
Mannheim, Germany
rehse@uni-mannheim.de

***Abstract*—Calculating event log measures (also known as features, metrics, and characteristics) is a common task required by many process mining applications. Process mining research studies and industrial applications often need to generate measures depending on their requirements. This has resulted in a plethora of event log measures being (re-)invented and (re-)implemented on different platforms. Fig4PM is an attempt toward building a standard, comprehensive, and reusable library for calculating event log measures. The current version of this open-source program offers 73 distinct control-flow measures either directly extracted from the literature (48 measures) or derived from the existing measures (25 measures). Eventually, our objective is to build a standard public Python library to facilitate feature generation in process mining applications.**

I. Introduction

Process mining projects typically start with extracting data from a process-aware information system and transforming them into an event log [1]. These event logs serve as input for virtually all process mining applications. In order to characterize the event logs and assess the specific differences (and similarities) among the traces, process analysts often employ event log measures, i.e., , "numeric representations of raw data" [2]. These measures can provide a priori insights about a log, which can then be used to draw conclusions about their properties. Typically, a measure is calculated at trace level and then aggregated to represent the event log characteristics. For example, calculating the length of each trace helps building the *average trace length* at event log level.

A wide range of process mining applications utilize such measures. We conducted a literature review to collect the studies that considered implementing new measures based on an event log's control-flow and found 21 scientific papers[1] ranging from 2001 to 2020. Interestingly, we noticed a certain level of overlap among these studies, i.e., different studies do not refer to fully-distinct and exclusive measures. According to the results of our literature review, many approaches require implementing measures, including (but not limited to) data preprocessing [3], data quality [4], predictive process mining [5], approaches that use deep learning techniques [6], business process simulation [7], process complexity analysis [8], and trace clustering [9].

To avoid the repeated (re-)invention and (re-)implemented of the same event log measures on different platforms, we introduce the Fig4PM library.[2] It provides researchers and practitioners with a basic library to access previously implemented event log measures and is specifically set out to be a starting point for ongoing development efforts. Prospective users may contribute to this project by developing new measures, improving the existing functions, add more data connectors, and improve its overall performance[3].

II. Measures

In Fig4PM, we distinguish two types of measures based on the underlying data structure. Linear measures perceive a trace as an array, matrix, or sequence of letters (a string), whereas non-linear measures perceive a trace as a directed graph, i.e., nodes represent activities while sequences determine edges [9].

A. Measures Derived From Linear Structures

Table I lists the measures derived from linear structures that were identified in the literature. For each measure, we list its abbreviation, description, and literature source. The measures are separated into groups based on their literature source and intended purpose.

The first two groups provide a brief overview of the log size and variability. The large third group measures structuredness and variance, i.e., risk of producing a Spaghetti model [13]. To quantify these properties, we can measure reoccurring behavior in terms of self-loops and repetitions as well as the number of start and end events which concern variability in initialization or termination. As more elaborate measures for structuredness, we measure the *number of distinct traces per 100 traces* ($tcpht$), *absolute trace coverage* (tco) and the *relative trace coverage* ($rtco$). The lower $tcpht$, the more structured the underlying event log. tco represents the minimum number of distinct traces required to cover 80% of all traces in the log hence, evaluating the variants' frequencies. Relating tco to ntc yields the relative trace coverage, which is better suited for comparison across different (sub-)logs.

The fourth group consists of several measures based on density, similarity (diversity), and complexity. The fifth group measures event log entropy using 3 different methods.

[1]Material is available at https://doi.org/10.6084/m9.figshare.14912313.v2.

[2]Code is available at https://github.com/f-zand/fig4pm.

[3]Demo video available at https://bit.ly/3iTt5MR.

TABLE I
LITERATURE-BASED MEASURES - LINEAR STRUCTURE

Abbreviation	Measure
ne	Total number of events [10]–[12]
nec	Total number of event classes [10]–[12]
nt	Total number of traces [10]–[12]
ntc	Total number of trace classes [10], [11]
atl	Average trace length [13], [14]
$mitl$	Minimum trace length [13], [14]
$matl$	Maximum trace length [13], [14]
ats	Avg. trace size (level of detail) [11], [12], [15]
$nsec$	Number of distinct start events [10], [13]
$ntec$	Number of distinct end events [10], [13]
$ntsl$	Abs. # traces with a self-loop [13]
ntr	Abs. # traces with a repetition [13]
$rnsec$	Rel.# distinct start events [13]
$rntec$	Rel. # distinct end events [13]
$rntsl$	Rel. # traces with a self-loop [13]
$rntr$	Rel. # traces with a repetition [13]
$anslt$	Avg. # self-loops per trace [13]
$manslt$	Max. # self-loops per trace [13]
$asslt$	Avg. size of self-loops per trace [13]
$masslt$	Max. size of self-loops per trace [13]
$tcpht$	# distinct traces per hundred traces [13]
tco	Absolute trace coverage [13]
$rtco$	Relative trace coverage [13]
edn	Event density [11], [15]
thr	Traces heterogeneity rate [11]
tsr	Trace similarity rate [11]
cf	Complexity factor [11]
std	Simple trace diversity [15]
atd	Advanced trace diversity [15]
$tentr$	Trace entropy [16]
$prentr$	Prefix entropy [16]
$abentr$	All-block entropy [16]

B. Measures Derived From Non-Linear Structure

Table II lists the literature-based measures derived from non-linear structures. In comparison to the linear measures, their number is rather limited. Many measures from the literature require post-discovery knowledge which is out of scope for this study. The remaining measures mainly focus on the directly-follows-graph (DFG) of the event log, quantifying the relationship between its nodes N and edges A.

TABLE II
LITERATURE-BASED MEASURES - NON-LINEAR STRUCTURE

Abbreviation	Measure
N	Number of nodes / vertices
A	Number of arcs / edges
$gcnc$	Coefficient of network connectivity [17], [18]
$gand$	Average node degree [17]
$gmnd$	Maximum node degree [17]
gdn	Density [17]
gst	Structure [12]
gcn	Cyclomatic number [18]
gdm	Graph diameter [17]
gcv	Number of cut vertices [14]
$gsepr$	Separability ratio [17]
$gseqr$	Sequentiality ratio [17]
gcy	Cyclicity [17]
gaf	Affinity [12]
$gspc$	Simple path complexity [19]

C. Self-Developed Measures

Inspired by the initial set of measures, we created 25 new measures to improve comprehensiveness and cover more topics. Linear structure includes measures focusing on frequency, connectedness, trace length, trace profile, and spatial proximity. Non-linear structure measures include measures based on modularity, cut-vertices, and activity labeling. In the additional material to this work, we provide a summary of the literature review and a list of all measures plus their respective formulas.

REFERENCES

[1] W. van der Aalst, "Process-aware information systems: Lessons to be learned from process mining," in *Transactions on Petri Nets and Other Models of Concurrency II*. Springer, 2009, pp. 1–26.

[2] A. Zheng and A. Casari, *Feature engineering for machine learning: principles and techniques for data scientists*. O'Reilly Media, 2018.

[3] M. Fani Sani, S. van Zelst, and W. van der Aalst, "Repairing outlier behaviour in event logs using contextual behaviour," *Enterprise Modelling and Information Systems Architectures*, vol. 14, 2019.

[4] S. Suriadi, R. Andrews, A. H. ter Hofstede, and M. T. Wynn, "Event log imperfection patterns for process mining: Towards a systematic approach to cleaning event logs," *Information systems*, vol. 64, pp. 132–150, 2017.

[5] A. E. Márquez-Chamorro, M. Resinas, and A. Ruiz-Cortés, "Predictive monitoring of business processes: a survey," *IEEE Transactions on Services Computing*, vol. 11, no. 6, pp. 962–977, 2017.

[6] F. Taymouri, M. La Rosa, S. Erfani, Z. D. Bozorgi, and I. Verenich, "Predictive business process monitoring via generative adversarial nets: The case of next event prediction," in *International Conference on Business Process Management*. Springer, 2020, pp. 237–256.

[7] N. Martin, B. Depaire, and A. Caris, "The use of process mining in business process simulation model construction," *Business & Information Systems Engineering*, vol. 58, no. 1, pp. 73–87, 2016.

[8] A. Augusto, J. Mendling, M. Vidgof, and B. Wurm, "The connection between process complexity of event sequences and models discovered by process mining," *arXiv preprint arXiv:2106.07990*, 2021.

[9] F. Zandkarimi, J.-R. Rehse, P. Soudmand, and H. Hoehle, "A generic framework for trace clustering in process mining," in *2020 2nd International Conference on Process Mining (ICPM)*. IEEE, 2020, pp. 177–184.

[10] J. Ribeiro, J. Carmona, M. Misir, and M. Sebag, "A recommender system for process discovery," in *Business Process Management*. Springer, 2014, pp. 67–83.

[11] M. Kherbouche, N. Laga, and P. Masse, "Towards a better assessment of event logs quality," in *Symposium Series on Computational Intelligence*. IEEE, 2 2017.

[12] C. Günther, "Process mining in flexible environments," Ph.D. dissertation, Eindhoven University of Technology, 2009.

[13] M. Swennen, G. Janssenswillen, M. Jans, B. Depaire, and K. Vanhoof, "Capturing Process Behavior with Log-Based Process Metrics," in *SIMPDA*, 2015, pp. 141–144.

[14] S. Van den Broucke, C. Delvaux, J. Freitas, T. Rogova, J. Vanthienen, and B. Baesens, "Uncovering the relationship between event log characteristics and process discovery techniques," in *Business Process Management*. Springer, 2014, pp. 41–53.

[15] M. Benner-Wickner, M. Book, T. Brückmann, and V. Gruhn, "Examining case management demand using event log complexity metrics," in *18th International Enterprise Distributed Object Computing Conference Workshops and Demonstrations*. IEEE, 2014, pp. 108–115.

[16] C. Back, S. Debois, and T. Slaats, "Towards an entropy-based analysis of log variability," in *Business Process Management Workshop*, vol. 308. Springer, 2018, pp. 53–70.

[17] J. Mendling, "Detection and Prediction of Errors in EPC Business Process Models," Ph.D. dissertation, Vienna University of Business Administration, 2007.

[18] A. Latva-Koivisto, "Finding a Complexity Measure for Business Process Models," Helsinki University of Technology, Tech. Rep., 2001.

[19] B. Pentland, P. Liu, W. Kremser, and T. Haerem, "The Dynamics of Drift in Digitized Processes," *MIS Quarterly*, vol. 44, no. 1, pp. 19–47, 1 2020.