Towards an Empirical Evaluation of Imperative and Declarative Process Mining

Christoffer Olling Back[1](✉), Søren Debois[2], and Tijs Slaats[1]

[1] Department of Computer Science, University of Copenhagen, Emil Holms Kanal 6, 2300 Copenhagen S, Denmark
{back,slaats}@di.ku.dk

[2] Department of Computer Science, IT University of Copenhagen, Rued Langgaards Vej 7, 2300 Copenhagen S, Denmark
debois@itu.dk

Abstract. Process modelling notations fall in two broad categories: declarative notations, which specify the *rules* governing a process; and imperative notations, which specify the *flows* admitted by a process. We outline an empirical approach to addressing the question of whether certain process logs are better suited for mining to imperative than declarative notations. We plan to attack this question by applying a flagship imperative and declarative miner to a standard collection of process logs, then evaluate the quality of the output models w.r.t. the standard model metrics of precision and generalisation. This approach requires perfect fitness of the output model, which substantially narrows the field of available miners; possible candidates include Inductive Miner and MINERful. With the metrics in hand, we propose to statistically evaluate the hypotheses that (1) one miner consistently outperforms the other on one of the metrics, and (2) there exist subsets of logs more suitable for imperative respectively declarative mining.

Keywords: Process mining · Modelling paradigms · Statistical evaluation · Declarative models · Imperative models · Hybrid models · Evaluation metrics

1 Introduction

Workflow notations are commonly categorised as falling within either the *imperative* or *declarative* paradigm [1]. Imperative notations use flow-based constructs to explicitly model the *paths* through a process [2]. Declarative notations use

T. Slaats—This work is supported by the Hybrid Business Process Management Technologies project (DFF-6111-00337) funded by the Danish Council for Independent Research, and the EcoKnow project (7050-00034A) funded by the Innovation Foundation.

constraint-based constructs to model the *rules* of a process. A declarative model allows all paths not forbidden by the constraints, and therefore the behaviour of the model is implicit in the rules and needs to be deduced by the system or users [3–5]. While the imperative paradigm is more mature, both paradigms have seen industrial adoption [6–8].

A recent trend in both academia and industry is to extract models from real-life data via *process discovery* [9], where an *output model* is automatically constructed from an *event log* of observed process executions. Research into this approach has focused primarily on the discovery of imperative models, but substantial energy has been directed towards algorithms that discover declarative models as well [10–12].

Thus, when constructing process models by process discovery, we have a choice regarding which paradigm to use. Does one approach return better models than the other? Would such a difference be universal or depend on the particular input log?

This paper outlines an approach to empirically evaluating the effect of miner paradigm (*independent variable*) on output model quality (*dependent variable*) [13,14]. We propose to measure model quality using notation-agnostic metrics for *precision* and *generalisation* from [15], which apply equally to imperative and declarative models. These metrics are analogous to the standard data mining metrics of the same name and are intended to capture the degree to which a model is underfitting or overfitting the data, respectively. Other quality metrics exist, such as fitness, simplicity/understandability, and soundness; but we are forced to keep these as *controlled variables* due to the fact that this formulations of precision and generalisation restricts our choice of miners, which in turn restricts our ability to include other metrics as dependent variables. Namely, since precision and generalisation require output models either be perfectly fitting or that data be aligned to the model to account for "noise", and since we have chosen to exclude the alignment procedure as a *confounding variable* in the first iteration of this evaluation approach (see Sect. 2.2), we are restricted to perfectly fitting output models.

We propose Inductive Miner [16] and MINERful [11] as representatives of the imperative and declarative paradigms, respectively, as they fulfill our requirements and are widely considered to be at the cutting edge of their respective fields. Evaluating the miners on publicly available, real-life logs, we test the following hypotheses:

Hypothesis 1: One miner consistently outperforms the other on one of the metrics:

(a) outperformance on precision (b) outperformance on generalisation

Hypothesis 2: There exist subsets of logs:

(a) more suited for imperative mining (b) more suited for declarative mining

That is, there exists a subset of logs which when mined either declaratively or imperatively represents a Pareto improvement over the other; *and* this deviation from the zero mean lies outside of the bounds of what can be accounted for by random chance.

A Pareto improvement simply denotes an improvement on at least one metric without sacrificing performance on the remaining metric. The zero mean is the mean of the probability distribution associated with the null hypothesis, and represents no performance difference between models produced by different miners from the same log.

Note that in the most extreme case, a subset may consist of a single log which is best suited to one paradigm, with the remaining logs showing only an insignificant difference in precision and generalisation, or requiring a tradeoff between the metrics, thus not a Pareto improvement. We are, in fact, testing multiple sub-hypotheses for hypothesis 2: one for each log. To compensate for the increased likelihood of making a type I error (false positive), we perform the appropriate adjustment to statistical significance testing: a *Bonferroni correction*. We leave as future work the task of identifying the characteristics of event logs which distinguish them as best suited for a given paradigm, in the interest of first rigorously establishing a clear framework for evaluation.

2 Methods

2.1 Log Selection

In the interest of reproducibility, we base log selection on the criteria of public availability, drawing upon the IEEE Task Force on Process Mining Real-life Event Log Collection[1], with the addition of one additional real-life log originating from our own industrial contact, the Dreyer Foundation in Denmark [17]. The logs stem from diverse sectors, including healthcare related processes, fine management, permit, loan and grant applications, as well as production, software engineering, and robotic vehicle related processes. The logs vary in degree of structure, number of activities, and trace length.

2.2 Process Discovery

We will mine the selected logs both imperatively and declaratively, selecting miners according to the following criteria:

1. Miners must be configurable to always produce perfectly fitting models.
2. Miners must be configurable to produce models of a given simplicity, save one which can serve as a benchmark.

The first criterion follows from both precision and generalisation requiring perfect fitness of the output model. It would have been an option to allow non-fitting output, and then use model-log alignment [15,18], but without domain

[1] http://data.4tu.nl/repository/collection:event_logs_real.

knowledge or access to an expert, we cannot know which exact alignment is more appropriate for the real-world log. This means that we would be evaluating not just the mining algorithm, but the combination of mining algorithm and alignment function. In particular, we would not know whether to attribute a result in favour of one miner over the other to the miner itself, or to a fortunate choice of alignment for that particular miner.

The second criterion follows partly from a tendency of declarative miners to produce output models containing excessive numbers of constraints: for large logs, on the order of *hundreds of thousands*. More importantly, we require model simplicity to be held constant, so that the choice of mining algorithm remains the only independent variable.

The two criteria left us only two miners: The Inductive Miner and MINERful.

Inductive Miner is an imperative miner developed by Leemans et al. [16] which uses a divide-and-conquer approach to generate sound, block-structured process models output as process trees or Petri nets. With only one parameter, noise threshold, which for our purposes must be held at 1.0 to ensure perfect fitness, the model generated by Inductive Miner provides a baseline model from which to set a threshold on model simplicity.

MINERful is a *declarative* miner developed by Di Ciccio et al. [11]. It uses a two-phase approach: in the first phase, a knowledge base of statistical information on the log is built; in the second, this knowledge is queried in order to infer the constraints of the process. The output is a Declare model, possibly including negative constraints.

MINERful has configurable thresholds for *support*, *interest factor*, and *confidence*. By iteratively adjusting these settings until a model is found which has the highest possible number of constraints without exceeding the complexity of the imperative model, we ensure that the imperative and declarative models are of comparable simplicity. We note that while many measures of simplicity have been proposed for imperative models, there exists no widely accepted method for comparing the simplicity of imperative and declarative models. For this reason, we begin by simply comparing the number of edge elements: edges between transitions vs. constraints between activities.

2.3 Computing Metrics

Defining standard measures for precision and generalisation remains an open research challenge for two main reasons. First, in process mining, data is generally not assumed to be labelled, i.e. event logs contain examples of what *did* happen, not what should *not* happen. This means that the standard definitions used in data mining and statistics cannot be applied to process discovery, since they rely on defining true and false positives, and true and false negatives. Second, the prevalence of unbounded loops in process models means that they often describe an infinite set of allowed behaviour. Therefore, definitions of precision and generalisation which take into account all of the of behaviour allowed by the model are not applicable in practice. Instead most metrics aim to reduce the measured behaviour of the model to a finite set of traces.

Metric Selection. To compare imperative and declarative models, we require metrics that can be applied to both equally. This means that they need to be defined on either the level of languages or transition systems. Accordingly, we have chosen to employ the metrics introduced in [15], in particular:

Precision [15, p. 10] measures the degree to which a model is "underfitting" or "allowing too much behaviour" relative to the input log. This particular metric is based on the notion of *escaping edges*, which represent a point at which the model allows behaviour not seen in the log. The measured amount of additional behaviour is kept finite by only considering the first divergent activity. I.e. an escaping edge may lead to a loop representing an infinite set of traces that did not occur in the log, but only the trace ending with the first divergent activity will be counted.

Generalisation [15, p. 11], on the other hand, measures the degree to which a model is "overfitting": is there behaviour not allowed by the model and not exhibited in the log, but that can be reasonably expected to occur in the future? This particular metric approximates generalisation by estimating for each state in the model the likelihood that a new, hitherto unseen, activity will occur. This estimation is based on the number of activities that have been observed, and how often the state was visited. Two alternatives are offered: *event-based generalisation* takes into account the number of visits to a state, *state-based generalisation* does not.

Implementation. The widely used process mining framework, ProM, contains a plugin for computing the metrics of [15] on Petri nets, but does not offer support for declarative models. Also, we seek the ability to run tests in batches and easily pipeline several operations (mining, metrics computation, analysis) on multiple logs. Therefore we developed our own evaluation framework[2]. The code, methods and results are straightforward to inspect and reproduce by following instructions provided on the associated wiki. The framework was tested against the examples and results reported in [15].

Challenges arise when computing precision and generalisation: mainly regarding time and space efficiency, but also handling nondeterminism arising from silent transitions present in models produced by Inductive Miner. When identifying enabled activities in a given marking, a greedy algorithm will naively follow silent transitions until encountering a non-silent transition, potentially firing silent transitions unnecessarily and associating incorrect markings with an event: subsequently excluding activities which should be enabled. Using the shortest path to a non-silent transition prevents this.

[2] Available at: https://bitbucket.org/coback/qmpm.

To minimize redundancy, a prefix tree is built from the event log, replaying each trace on the given model as it is added to the trie. In each node (corresponding to an event in the log), the state of the model is saved, unless the node has been visited previously, in which case a counter associated with the node is incremented, recording the number of occurrences of that prefix. Finally, a map containing model states (markings) as keys, and sets of nodes (events) as values, is maintained in order to facilitate the calculation of state-based generalisation. Given an event, the enabled activities in the log simply correspond to that node's children, while the enabled activities in the model are obtained by querying the model using the model state associated with that node. This approach minimizes redundancies, keeping state-space enumeration to a minimum.

3 Conclusion

We outline an approach to systematically compare the performance of imperative and declarative process mining algorithms based on notation-agnostic quality metrics for precision and generalisation defined in [15]. We will investigate two hypotheses: first, one miner performs better on precision and/or generalisation; second, there exist logs on which either miner provides a statistically significant Pareto improvement.

To the best of our knowledge, this will be the first study comparing imperative and declarative process discovery techniques using this approach. Future evaluations incorporating other aspects of the process mining life-cycle, e.g. alignment, will be able to use this approach as a point of reference. Not least, we contribute a comprehensive software framework and tackle a number of methodological and implementation challenges, providing a foundation upon which further work can build.

Finally, we believe that the proposed study will be extremely valuable to the field of hybrid process mining [19–21], which aims to combine the strengths of the two paradigms. Research into which characteristics identify a portion of a log as more suitable to one paradigm have been hampered by the lack of an objective procedure on which to compare models across paradigms [22]. Our approach lays the groundwork for addressing this shortcoming.

References

1. Reijers, H.A., Slaats, T., Stahl, C.: Declarative modeling–an academic dream or the future for BPM? In: Daniel, F., Wang, J., Weber, B. (eds.) BPM 2013. LNCS, vol. 8094, pp. 307–322. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40176-3_26
2. Van der Aalst, W.M.P.: Verification of workflow nets. In: Azéma, P., Balbo, G. (eds.) ICATPN 1997. LNCS, vol. 1248, pp. 407–426. Springer, Heidelberg (1997). https://doi.org/10.1007/3-540-63139-9_48
3. van der Aalst, W.M.P., Pesic, M., Schonenberg, H., Westergaard, M., Maggi, F.M.: Declare. Webpage (2010). http://www.win.tue.nl/declare/

4. Debois, S., Hildebrandt, T.T., Slaats, T.: Replication, refinement & reachability: complexity in dynamic condition-response graphs. Acta Informatica **55**, 489–520 (2017)
5. Hull, R., et al.: Business artifacts with guard-stage-milestone lifecycles. In: DEBS 2011, pp. 51–62 (2011)
6. Object Management Group: Business Process Modeling Notation Version 2.0. Technical report, Object Management Group Final Adopted Specification (2011)
7. Marquard, M., Shahzad, M., Slaats, T.: Web-based modelling and collaborative simulation of declarative processes. In: Motahari-Nezhad, H.R., Recker, J., Weidlich, M. (eds.) BPM 2015. LNCS, vol. 9253, pp. 209–225. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-23063-4_15
8. Object Management Group: Case Management Model and Notation, version 1.0. Webpage, May 2014. http://www.omg.org/spec/CMMN/1.0/PDF
9. Van der Aalst, W.M.P.: Process Mining: Data Science in Action. Springer, Heidelberg (2016)
10. Maggi, F.M., Bose, R.P.J.C., van der Aalst, W.M.P.: Efficient discovery of understandable declarative process models from event logs. In: Ralyté, J., Franch, X., Brinkkemper, S., Wrycza, S. (eds.) CAiSE 2012. LNCS, vol. 7328, pp. 270–285. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-31095-9_18
11. Di Ciccio, C., Mecella, M.: On the discovery of declarative control flows for artful processes. ACM Trans. Manag. Inf. Syst. **5**(4), 24 (2015)
12. Debois, S., Hildebrandt, T.T., Laursen, P.H., Ulrik, K.R.: Declarative process mining for DCR graphs. In: Proceeding of the Symposium on Applied Computing, SAC 2017, pp. 759–764 (2017)
13. Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: On the role of fitness, precision, generalization and simplicity in process discovery. In: Meersman, R., et al. (eds.) OTM 2012. LNCS, vol. 7565, pp. 305–322. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33606-5_19
14. Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: Quality dimensions in process discovery: the importance of fitness, precision, generalization and simplicity. Int. J. Coop. Inf. Syst. **23**(1), 1440001 (2014)
15. van der Aalst, W.M.P., Adriansyah, A., van Dongen, B.F.: Replaying history on process models for conformance checking and performance analysis. Wiley Interdisc. Rew. Data Min. Knowl. Disc. **2**(2), 182–192 (2012)
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). https://doi.org/10.1007/978-3-642-38697-8_17
17. Debois, S., Slaats, T.: The analysis of a real life declarative process. In: CIDM 2015, pp. 1374–1382 (2015)
18. Adriansyah, A., Munoz-Gama, J., Carmona, J., van Dongen, B.F., van der Aalst, W.M.P.: Alignment based precision checking. In: La Rosa, M., Soffer, P. (eds.) BPM 2012. LNBIP, vol. 132, pp. 137–149. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36285-9_15
19. Slaats, T., Schunselaar, D.M.M., Maggi, F.M., Reijers, H.A.: The semantics of hybrid process models. In: Debruyne, C. (ed.) OTM 2016. LNCS, vol. 10033, pp. 531–551. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-48472-3_32
20. Maggi, F.M., Slaats, T., Reijers, H.A.: The automated discovery of hybrid processes. In: Sadiq, S., Soffer, P., Völzer, H. (eds.) BPM 2014. LNCS, vol. 8659, pp. 392–399. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10172-9_27

21. Schunselaar, D.M.M., Slaats, T., Maggi, F.M., Reijers, H.A., van der Aalst, W.M.P.: Mining hybrid business process models: a quest for better precision. In: Abramowicz, W., Paschke, A. (eds.) BIS 2018. LNBIP, vol. 320, pp. 190–205. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93931-5_14
22. Back, C.O., Debois, S., Slaats, T.: Towards an entropy-based analysis of log variability. In: Teniente, E., Weidlich, M. (eds.) BPM 2017. LNBIP, vol. 308, pp. 53–70. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-74030-0_4