

LogPPL: A Tool for Probabilistic Process Mining^{*}

Martin Kuhn¹, Joscha Grüger², Christoph Matheja³ and Andrey Rivkin³

¹German Research Center for Artificial Intelligence (DFKI), SDS Branch Trier, Trier, Germany

²University of Trier, Germany

³Technical University of Denmark, Kgs. Lyngby, Denmark

Abstract

This paper introduces LogPPL, a novel tool designed to bridge the gap between Data Petri Nets (DPNs) and probabilistic programming, enabling the generation of event logs with statistical guarantees via probabilistic program executions. LogPPL implements the transformation of DPNs into probabilistic programs written in the WebPPL language, allowing to harness the power of simulation and inference engines supplied for the WebPPL environment. Our tool simplifies the configuration of the DPN simulation setup and allows for exporting both event logs in XES format as well as WebPPL files. LogPPL capabilities are demonstrated through various scenarios, showcasing its potential to enhance process mining tasks by offering rigorous statistical modeling and advanced simulation features. The tool's design, features, and performance are evaluated, highlighting its utility in both academic and industrial settings.

Keywords

Process Mining, Data Petri Nets, Probabilistic Programming, Event Log Generation, WebPPL, Statistical Simulation, DPN Simulation

Metadata description	Value
Tool name	LogPPL
Current version	0.1
Legal code license	GPL-3.0
Languages, tools and services used	Python, WebPPL, Docker
Supported operating environment	Microsoft Windows, GNU/Linux, Mac OS
Source code repository	https://github.com/martinkuhn94/LogPPL
Screencast video	https://github.com/martinkuhn94/LogPPL/tree/main/examples/screencast

1. Introduction

The proliferation of data-driven approaches in Business Process Management (BPM) and Process Mining (PM) has led to a significant reliance on data Petri nets (DPNs) – a Petri net-based formalism which combines classical P/T-nets with guarded reasoning about bounded memory


ICPM 2024 Tool Demonstration Track, October 14-18, 2024, Kongens Lyngby, Denmark

^{*}Corresponding author.

[†]These authors contributed equally.

✉ martin.kuhn@dfki.de (M. Kuhn); grueger@uni-trier.de (J. Grüger); chmat@dtu.dk (C. Matheja); ariv@dtu.dk (A. Rivkin)

ORCID: [0000-0002-3242-1251](https://orcid.org/0000-0002-3242-1251) (M. Kuhn); [0000-0001-7538-1248](https://orcid.org/0000-0001-7538-1248) (J. Grüger); [0000-0001-9151-0441](https://orcid.org/0000-0001-9151-0441) (C. Matheja); [0000-0001-8425-2309](https://orcid.org/0000-0001-8425-2309) (A. Rivkin)

 © 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

(which is also referred to as process or case variables) [1, 2]. DPNs serve as a powerful formalism for modeling and analyzing complex processes, supporting various tasks such as process discovery [1, 3], conformance checking [1, 3, 4, 5], formal verification [6, 7], and strategy synthesis [8]. Despite the strengths of DPNs, traditional simulation techniques used within this context have often lacked rigorous statistical foundations, especially when addressing stochastic processes [5, 9]. This gap underscores the need for more robust approaches that can incorporate statistical guarantees in the simulation and analysis of DPNs.

Probabilistic Programming [10, 11] (PP for short) offers a promising paradigm that could address these limitations by providing a systematic way to model statistical guarantees and perform inference in complex systems. The recent work “*Data Petri Nets Meet Probabilistic Programming*” [12] explores this intersection by proposing a novel approach which translates DPNs into probabilistic programs and uses the latter as net simulation engines. This approach proposed in [12] leverages the inherent capabilities of PP languages, such as WebPPL [13], Stan [14] and PyMC3 [15] to provide a statistically grounded simulation framework that integrates seamlessly with existing DPN models, and that can be used in such tasks as trace generation.

In this context, we introduce LogPPL – a comprehensive tool designed to bridge the gap between DPNs and probabilistic programming that enables event log generation with statistical guarantees. LogPPL implements the approach proposed in [12], offering an intuitive interface for converting DPN models into WebPPL programs, configuring various simulation parameters and analyzing the results. This tool attempts at simplifying the application of Probabilistic Programming (PP) engines in the process mining context, making them accessible by a potentially broader audience. Specifically, through LogPPL, users can harness the power of PP in such tasks as (synthetic) event log generation and “what-if” analysis.

This paper presents the design and functionality of LogPPL, demonstrating its utility in supporting advanced simulation and analysis tasks in BPM and PM. The tool’s implementation and its application in various scenarios is showcased, which highlights its potential to enhance the rigor and robustness of DPN-based simulations.

2. Innovations and Features

In [12], an approach was introduced that enables the use of probabilistic programming for simulation and reasoning about data-aware processes. The approach proposed a systematic translation of a data Petri net into a program written in a probabilistic programming language (which, in turn, is supported by most probabilistic programming systems). It was also demonstrated how resulting programs can be used to generate multi-perspective event logs with statistical guarantees.

DPNs extend traditional P/T-nets with transition guards that can manipulate scalar case variables. Each guard is split into a pre- and post-condition expressions, where the latter can define variable updates using primed copies of case variables. To make such nets simulatable and suitable for translation into PP programs, [12] proposes to equip the nets with schedulers for resolving non-deterministic choices. In particular, schedulers assign probability distributions to transitions and primed variables in guards, which in turn provide the desired statistical guarantees for simulating DPNs.

Setting up schedulers and translating them together with corresponding DPNs into “ready-to-simulate” PP programs is a complicated, error-prone, and time-consuming task. Thus, we propose LogPPL which fully automates the process of scheduler configuration for a given DPN, provides various predefined property-driven simulation setups, and automatically translates such configuration together with the related model into a PP program in WebPPL. The tool thus enables the generation of event logs with aforementioned statistical guarantees by performing internal calls to the MCMC (Markov Chain Monte Carlo) statistical inference engine of WebPPL.

The core features of the tool are:

- **Loading and visualizing DPN:** LogPPL initially loads the data Petri net (provided in PNML¹ format) and visualizes it together with all its guards. This visualization particularly aids in configuring the scheduler in subsequent steps.
- **Configuration of simulation parameters:** First, the tool offers predefined simulation setups, which essentially define properties that target the simulation process. One of such properties is the final marking reachability, which is often desirable in the context of event log generation. One may also set up custom properties that are defined on events produced by the simulated net. For example, one may define a property ensuring that x is at least 17.5 and t_3 was fired at most twice ($Observe(x > 17.5 \wedge \#t_3 \leq 2)$). Another two simulation parameters are the sample size which determines the number of traces to be generated, and the simulation length size which controls the maximum number of transition firings per simulation run.
- **Configuration of the scheduler:** For each prime variable and each transition that contains it, probability distributions needed to resolve non-determinism during simulation runs need to be defined. The tool offers 23 pre-defined distribution functions (Bernoulli, Beta, Binomial, Categorical, Cauchy, Delta, DiagCovGaussian, Dirichlet, Discrete, Exponential, Gamma, Gaussian, KDE, Laplace, LogisticNormal, Mixture, Multinomial, MultivariateBernoulli, MultivariateGaussian, Poisson, RandomInteger, TensorGaussian, TensorLaplace, Uniform) together with the distribution that can be manually configured. To assist with the process of configuring distributions and their parameters, the tool provides comprehensive assistance. An example configuration can be seen in Figure 1.
- **Translation and running simulation:** After the configuration step, the input DPN is translated into a probabilistic program that, apart from the net, takes into account the configuration of simulation parameters. The translation follows the procedure described in [12]. The following simulation process upon initiation invokes internally the WebPPL MCMC inference engine.
- **Event log & WebPPL export:** After the simulation, the event log, which conforms to the defined probability distributions, can be exported in the XES format. Furthermore, the WebPPL representation of the given Data Petri Net (DPN) can be downloaded.

These features make LogPPL a powerful tool for various process mining tasks. Synthetic event logs of size n are generated by executing the underlying WebPPL program simulating a given DPN for at least n times. Additionally, the same WebPPL program can be used to study (using tools offered by the WebPPL implementation) the distribution of process runs and explore

¹<https://www.pnml.org/>

conditional probabilities, thus enhancing understanding of the dynamics within a given data Petri net. Such capabilities facilitate advanced what-if analyses and the investigation of rare events, demonstrating PP’s flexibility and depth in modeling complex stochastic systems.

Configuration

Uploaded File: simple_auction.pnml

Scheduler Configuration [?](#)

Event: *init*

Guard: $((t > 0) \& \& (o' = 0))$

t' :

A continuous distribution where all outcomes in the interval [a, b] are equally likely.

Lower bound:

Upper bound:

Event: *bid*

Guard: $((t > 0) \& \& (o' > 0))$

o' :

A continuous distribution defined by a mean and standard deviation.

Mean:

Standard deviation:

Figure 1: Scheduler configuration options in the LogPPL tool include settings for guards with prime variables. In this context, *init* defines the initialization of a simple auction process, where the time variable t' is set based on a uniform distribution. The *bid* events generate the price o' following the Gaussian distribution.

3. Tool Maturity

LogPPL is a fully functional, stand-alone tool that is ready to be used by researchers and practitioners to convert DPNs into probabilistic programs and use the latter to generate event logs with statistical guarantees. The tool is designed with ease of use and accessibility in mind, ensuring that it can be adopted across a wide range of use cases and user expertise levels.

LogPPL supports Windows and Linux-based operating systems, and is packaged for installation via Docker, which simplifies the setup process and reduces potential compatibility issues. From a technical perspective, LogPPL is implemented as a web application utilizing Python and the Flask framework as its backend. The source code is released under the GPL-3.0 license.

The tool’s maturity is further demonstrated by its evaluation through a proof-of-concept, which was conducted to assess both its correctness and efficiency using tool performance indicators. To demonstrate its practical applicability, the approach was tested on two distinct DPNs: the Road Fine DPN [1], which includes 9 places, 19 transitions, 11 guards, and 8 variables, and the more complex Melanoma DPN [16], which consists of 50 places, 76 transitions, 52 guards, and 26 variables. In these evaluations, WebPPL’s MCMC inference engine was employed with various parameter settings, providing insights into the tool’s performance in different scenarios. The runtime across multiple simulation runs was measured, under different parameter combinations. Specifically, parameters such as run lengths (determined by the simulation loop parameter) ranging from 10 to 50 and the number of samples which ranges from 100 to 819,200 were tested, with a 180-second timeout for each run. Figure 2 displays the runtime over five computation cycles, where the number of generated runs doubles at each step, leading to an exponential increase in runtime. The resulting runtimes are within a reasonable timeframe for applications in real life scenarios, but still indicating potential for optimizations.

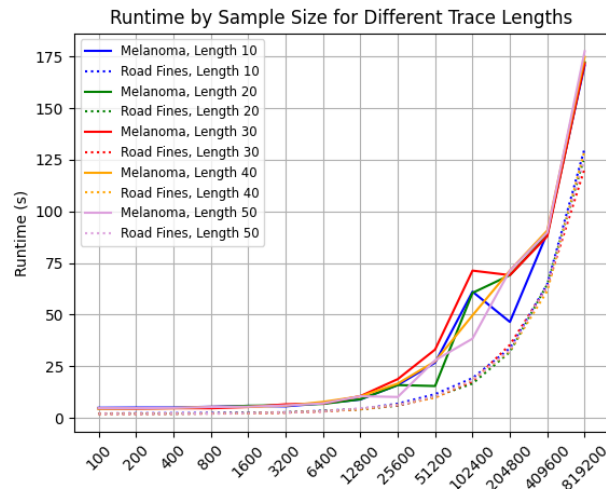


Figure 2: Evaluation results for Melanoma and Road Fine, highlighting the runtime (in seconds) in relation to the number of generated runs (x-axis) [12]

4. Conclusion and Future Work

In this paper, we introduce LogPPL – a novel tool that bridges the gap between DPNs and probabilistic programming, enabling the generation of event logs with statistical guarantees. By leveraging the power of the WebPPL PP language, LogPPL facilitates the translation of DPNs into probabilistic programs, allowing researchers and practitioners to simulate and analyze complex processes with respect to statistical guarantees on simulated net runs. The tool’s capabilities are particularly valuable for fields such as Business Process Management and Process Mining, where accurate modeling and simulation of process behaviors are essential for decision-making and process optimization. The tool’s design prioritizes usability and accessibility, providing a user-friendly interface to facilitate adoption by a broad audience.

While LogPPL represents a significant advancement in the simulation and analysis of DPNs, there are several possibilities for future work. One promising direction is the extension of LogPPL to support a wider range of process modeling languages like BPMN. This could serve an even broader audience and provide more versatility in the types of process models that can be simulated and analyzed.

Another potential enhancement involves improving the tool’s user interface to include advanced statistical visualizations of the simulation results. This could involve integrating dashboards or interactive charts that allow users to explore the distribution of process attributes across generated event logs. Such features can improve the interpretability of the simulation outcomes and also provide users with more insights into the dynamics of the simulated processes.

Additionally, future work could focus on automating the configuration of probabilistic models based on existing event log data. Developing methods that can automatically infer appropriate probabilistic models and parameters from historical data could significantly reduce the manual effort required to set up simulation parameters and increase the accuracy of the results. This

automation would make LogPPL even more accessible to users who may not have a deep understanding of probabilistic programming, but still wish to leverage its capabilities.

References

- [1] F. Mannhardt, M. de Leoni, H. A. Reijers, W. M. P. van der Aalst, Balanced multi-perspective checking of process conformance, *Computing* 98 (2016).
- [2] M. de Leoni, P. Felli, M. Montali, A holistic approach for soundness verification of decision-aware process models, in: *ER*, volume 11157 of *LNCS*, Springer, 2018, pp. 219–235.
- [3] F. Mannhardt, Multi-perspective Process Mining, Ph.D. thesis, TU/e, 2018.
- [4] P. Felli, A. Gianola, M. Montali, A. Rivkin, S. Winkler, Cocomot: Conformance checking of multi-perspective processes via SMT, in: A. Polyvyanyy, M. T. Wynn, A. V. Looy, M. Reichert (Eds.), *Proc. of BPM 2021*, volume 12875 of *LNCS*, Springer, 2021, pp. 217–234.
- [5] P. Felli, A. Gianola, M. Montali, A. Rivkin, S. Winkler, Conformance checking with uncertainty via SMT, in: *Proc. of BPM 2022*, LNCS, Springer, 2022.
- [6] P. Felli, M. Montali, S. Winkler, Soundness of data-aware processes with arithmetic conditions, in: X. Franch, G. Poels, F. Gailly, M. Snoeck (Eds.), *Proc. of CAiSE 2022*, volume 13295 of *LNCS*, Springer, 2022, pp. 389–406.
- [7] P. Felli, M. Montali, S. Winkler, Ctl^* model checking for data-aware dynamic systems with arithmetic, in: J. Blanchette, L. Kovács, D. Pattinson (Eds.), *Proc. of IJCAR 2022*, volume 13385 of *LNCS*, Springer, 2022, pp. 36–56.
- [8] M. de Leoni, P. Felli, M. Montali, Strategy synthesis for data-aware dynamic systems with multiple actors, in: *KR*, 2020, pp. 315–325.
- [9] F. Mannhardt, S. J. J. Leemans, C. T. Schwanen, M. de Leoni, Modelling data-aware stochastic processes - discovery and conformance checking, in: L. Gomes, R. Lorenz (Eds.), *Proc. of PETRI NETS 2023*, LNCS, Springer, 2023.
- [10] J.-W. van de Meent, B. Paige, H. Yang, F. Wood, An introduction to probabilistic programming, *arXiv preprint arXiv:1809.10756* (2018).
- [11] S. Russell, P. Norvig, *Artificial Intelligence, Global Edition A Modern Approach*, Pearson Deutschland, 2021.
- [12] M. Kuhn, J. Grüger, C. Matheja, A. Rivkin, Data petri nets meet probabilistic programming, in: A. Marrella, M. Resinas, M. Jans, M. Rosemann (Eds.), *Proc. of BPM 2024*, volume 14940 of *LNCS*, Springer, 2024.
- [13] N. D. Goodman, A. Stuhlmüller, *The Design and Implementation of Probabilistic Programming Languages*, <http://dippl.org>, 2014. Accessed: 2024-3-8.
- [14] B. Carpenter, A. Gelman, M. D. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. A. Brubaker, J. Guo, P. Li, A. Riddell, Stan: A probabilistic programming language, *Journal of statistical software* 76 (2017).
- [15] J. Salvatier, T. V. Wiecki, C. Fonnesbeck, Probabilistic programming in python using pymc3, *PeerJ Computer Science* 2 (2016) e55.
- [16] J. Grüger, T. Geyer, M. Kuhn, S. Braun, R. Bergmann, Verifying guideline compliance in clinical treatment using multi-perspective conformance checking: A case study, in: *Process Mining Workshops*, Springer, 2022.