$\text{Rims}_{\text{Tool}}$: a Hybrid Simulator for Business Processes

Francesca Meneghello[1,2,*], Chiara Di Francescomarino[3,†] and Chiara Ghidini[1,†]

[1]Fondazione Bruno Kessler, Trento, Italy
[2]Sapienza University of Rome, Rome, Italy
[3]DISI, University of Trento, Italy

Abstract

Business Process Simulation represents a powerful instrument for business analysts when analyzing and comparing business processes. Most of the state-of-the-art business process simulators, however, rely on Discrete event simulation, which requires various unrealistic assumptions and simplifications to perform experiments. Predictive Process Monitoring, on the other hand, offers a viable way to complete ongoing traces or to generate entire traces from scratch, via predictions of the next activities and their attributes. Predictive models, though, are usually based on black-box approaches that make it difficult to reason on what-if scenarios. $\text{RIMS}_{\text{Tool}}$ is a hybrid business process simulator that aims at combining predictive models built from data and Discrete event simulation at runtime in a white-box manner. The proposed tool, thus, is able to exploit the strengths and avoid the limitations of both approaches.

Keywords

Business Process Simulation, Machine Learning, Hybrid Simulation

1. Introduction

Business Process Simulation (BPS) refers to techniques for the simulation of business process behaviours and allows analysts to compare alternative scenarios and contribute to the analysis and improvement of business processes. Several tools exist both in academia and in the commercial world. examples include BIMP[1], CPNtool [1] and PLG [2]. If we focus on the technique used in simulation, the most widely used simulation technique in the context of BPS is Discrete Event Simulation (DES). DES simulators generate events based on the rules defined in the simulation model. Each event occurs at a specific time and indicates a change in the state of the system. However, defining simulation models require making unrealistic or oversimplified assumptions due to the expressive limitations of DES simulators. For example, in BIMP, the processing time of an activity can only be defined as a random probability distribution, instead of considering also previously performed activities, assigned resources, the current timestamp, and/or other event attributes. An alternative way to generate simulated runs by completing trace prefixes or

ICPM
*Corresponding author.
†These authors contributed equally.
fmeneghello@fbk.eu (F. Meneghello); c.difrancescomarino@unitn.it (C. Di Francescomarino); ghidini@fbk.eu (C. Ghidini)
0009-0008-8000-1770 (F. Meneghello); 0000-0002-0264-9394 (C. Di Francescomarino); 0000-0003-1563-4965 (C. Ghidini)

[1]BIMP simulator: https://bimp.cs.ut.ee/

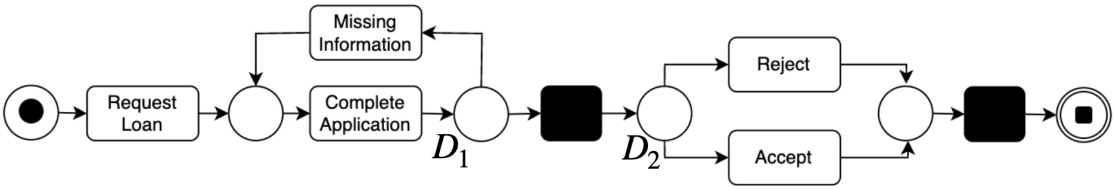

Figure 1: Petri net model describing a loan application handling process of a bank.

generating traces from scratch is offered by recent Deep Learning (DL) techniques applied to predictive process monitoring [3]. These models are extremely powerful in learning patterns that characterize the exact relationship between different trace elements – especially related to the temporal perspective, as differently from DES simulators, they do not need to make strong assumptions. However, unlike DES simulation models, DL models are problematic to use for analyzing and improving business processes, because of their black-box nature.

To overcome this issue, state-of-the-art techniques in BPM combine DL and DES in a **post-integration** fashion. *Dsim* [4] is the first attempt in which an entire simulation is performed, and then a DL model is used to add waiting and processing times to the events produced by the simulation model. In a recent paper [5] we presented Rims (Runtime Integration of Machine Learning and Simulation), an approach that provides a tight integration of the predictions of the DL model at **runtime** during the simulation. That work shows how the runtime integration allows us to fully exploit the predictions on specific process perspectives to improve the overall performance with respect to using the individual techniques separately or the post-integration approach.

In this demo paper we present $\text{RIMS}_{\text{Tool}}$, a hybrid simulator tool based on the Rims approach proposed in [5]. In comparison to the work presented in [5], the version of $\text{RIMS}_{\text{Tool}}$ described here further extends the potentiality of the runtime integration by adding time-related attributes (processing and waiting time predictions) to the control-flow perspective, i.e. for the prediction of the most likely branch in the decision point of a Petri net. Furthermore, $\text{RIMS}_{\text{Tool}}$ is a highly configurable tool that facilitates the integration of any predictive model in one or more process perspectives and/or the application of various configurations even on the same perspective (see Section 3).

2. Motivating Example

Let us consider the Petri net model in Figure 1, where a client asks for a loan, and the bank decides whether to provide it or not. To create a simulation model it is necessary to specify simulation parameters related to time, resources and control perspectives. For example, in DES simulators, arrival rates, processing and waiting times are typically approximated with fixed times or probability distributions (e.g. exponential, normal), while probabilities or condition rules are used for selecting a branch at decision points. This means that, for instance, (i) the customer arrival rate does not take into account seasonal behaviours and potential peaks; (ii) the processing time of activities as Complete Application does not take into account specific

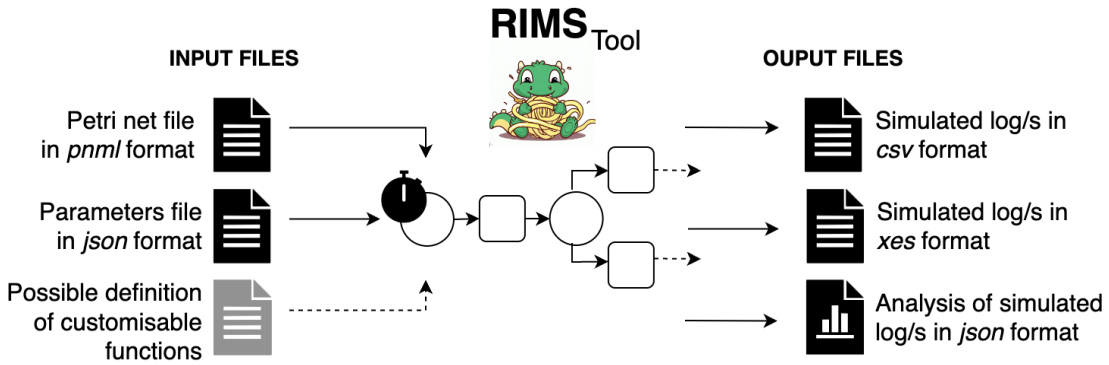

Figure 2: Structure of $\text{RIMS}_{\text{Tool}}$ defined as input and output files.

characteristics of the customer, i.e., whether he is new or not, or if he has to bring more documents to ensure solvency; (iii) the waiting time between Complete Application and the decision point ($\mathcal{D}_2$) and (iv) the choice of the branch to follow at $\mathcal{D}_2$, i.e., whether the bank decides to accept or reject the loan, do not take into account the characteristics of the applicant, e.g., her reliability.

$\text{RIMS}_{\text{Tool}}$ allows us to avoid several approximations and unrealistic behaviors such as those adopted to simulate the process in Figure 1 with a DES simulator. In $\text{RIMS}_{\text{Tool}}$, we can integrate a time series model to simulate a more realistic client request arrival rate. Furthermore, it can incorporate at runtime two predictive models for processing and waiting times. In this way, the timing of a process instance depends on the client it refers to. Finally, a predictive model for $\mathcal{D}_2$ prevents the generation of a simulated trace in which an insolvent client gets the loan from the bank.

3. RIMS: Innovation and Features

As shown in Figure 2, $\text{RIMS}_{\text{Tool}}$ takes as input a Petri net process model[2] and a set of simulation parameters in order to generate the simulation model. These two files are enough for $\text{RIMS}_{\text{Tool}}$ to generate a DES simulation. The DES simulator can be transformed into a hybrid one, by defining one or more customisable functions that allow to leverage any predictive models in the dedicated configuration file $custom\_functions.py$. Specifically, the user can define six customisable functions to manage different process perspectives. Two custom functions (*case_function_attribute* and *event_function_attribute*), allow for specifying any case and event attributes (e.g., the requested loan amount in the example). Through the other four customisable functions (*custom_arrivals_time, custom_processing_time, custom_waiting_time*, custom_decision_mining), one or more predictive models can be integrated into the simulator to predict respectively: (i) the arrival time of a new trace; (ii) the processing time of an activity; (iii) the waiting time between two activities and (iv) the process path from a decision point.

[2] $\text{RIMS}_{\text{Tool}}$ does not accept a BPMN model as input, however, it can be easily transformed into a Petri net, through the *PM4Py* Python library.

To compute the latter three types of predictions the custom functions take as input parameter also intra-case and inter-case features[3] of the running event. For example, the prediction of the waiting time between the Complete Application and Accept activity is possibly based on intra-case features, such as the current timestamp and the assigned resource, and inter-case features, such as the resource's queue and the current number of ongoing traces. Moreover, $\text{RIMS}_{\text{Tool}}$ is rather flexible and allows us to use different configurations also for the same perspective (e.g., fixed processing time for an activity and predicted time for another one).

At the end of the simulation, $\text{RIMS}_{\text{Tool}}$ returns as output the files shown in Figure 2, i.e., the simulation log in the *csv* and *xes* formats, and a brief analysis of the simulation (e.g., number of traces and events generated, process resource usage, etc.).

4. Maturity and Conclusion

$\text{RIMS}_{\text{Tool}}$ is a Python-based tool implemented by leveraging the Simpy[4] and the PM4Py library[5]. The documentation on how to install and run the tool, as well as a video and the code of $\text{RIMS}_{\text{Tool}}$ are available at the Github repository[6]. Three different case studies describing the integration of Random Forest models – focused on the time and control-flow predictions – within a DES model are also available in the documentation. The maturity of the tool is further demonstrated in the more complex case studies described in [5] and in [6], where an ad-hoc version of $\text{RIMS}_{\text{Tool}}$ is used to simulate the behavior of the environment in response to the agent's action recommended by the discovered optimal policy in a Reinforcement Learning scenario.

$\text{RIMS}_{\text{Tool}}$ is a hybrid BPS simulator able to combine DES and predictive models at runtime in a white-box manner. As future work, we want to generalize and extend it, providing users with the possibility to integrate predictive models also for the resource perspective, as well as with the capability to simulate only the activities of an actor in response to the activities performed by other actors. well.

Acknowledgments

We acknowledge the support of the PNRR project FAIR - Future AI Research (PE00000013), under the NRRP MUR program funded by the NextGenerationEU.

References

[1] M. Westergaard, L. M. Kristensen, The access/cpn framework: A tool for interacting with the CPN tools simulator, in: G. Franceschinis, K. Wolf (Eds.), Applications and Theory of Petri Nets, 30th International Conference, PETRI NETS 2009, Paris, France, June 22-26, 2009. Proceedings, volume 5606 of *Lecture Notes in Computer Science*, Springer, 2009, pp. 313–322.

[3] All intra-case and inter-case features are listed in the documentation.
[4] https://simpy.readthedocs.io/en/latest/
[5] https://pm4py.fit.fraunhofer.de/
[6] https://github.com/francescameneghello/RIMS_tool

[2] A. Burattin, PLG2: multiperspective process randomization with online and offline simulations, in: L. Azevedo, C. Cabanillas (Eds.), Proceedings of the BPM Demo Track 2016 Co-located with the 14th International Conference on Business Process Management (BPM 2016), Rio de Janeiro, Brazil, September 21, 2016, volume 1789 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2016, pp. 1–6.

[3] C. Di Francescomarino, C. Ghidini, Predictive process monitoring, in: Process Mining Handbook, volume 448 of *LNBIP*, Springer, 2022, pp. 320–346.

[4] M. Camargo, M. Dumas, O. G. Rojas, Learning accurate business process simulation models from event logs via automated process discovery and deep learning, in: Advanced Information Systems Engineering - Int. Conf., CAiSE 2022, Proc., volume 13295 of *LNCS*, Springer, 2022, pp. 55–71.

[5] F. Meneghello, C. Di Francescomarino, C. Ghidini, Runtime integration of machine learning and simulation for business processes, in: Proc. of the 5th Int. Conference on Process Mining (ICPM 2023), 2023. To appear.

[6] S. Branchi, A. Buliga, C. Di Francescomarino, C. Ghidini, F. Meneghello, M. Ronzani, Recommending the optimal policy by learning to act from temporal data, arXiv preprint arXiv:2303.09209 (2023).